Universidad Politécnica de Cartagena

industriales
etsii UPCT

Comunicación y control de las articulaciones de un exoesqueleto

Titulación: Ingeniero Industrial
Alumno/a: José Antonio Parra Aznar
Director/a/s: Dr. Javier Molina Vilaplana
Dr. Jorge Feliu Batlle

Cartagena, 18 de Diciembre de 2015

Agradecimientos

En primer lugar, y como personas más importantes en mi desarrollo personal y profesional, agradecer a mi padre, a mi madre y a mi hermana. Estas personas me han ayudado, han apoyado y soportado toda la carga que supone realizar una formación superior, y en concreto, este proyecto. Mencionar también a D. Pedro A. Parra, una persona que me ha apoyado en todo lo relacionado a mi formación y apoyado en todo lo referente a este proyecto. No quisiera dejar de agradecer a una persona muy importante en mi vida, una persona que siempre estará en mi recuerdo y que ha sido un gran apoyo todos estos años, Gracias Abuela.

Agradecer al Dr. Javier Molina Vilaplana su tiempo y dedicación como director de este proyecto, así como sus consejos, apoyo y motivación recibida durante el proyecto. Agradecer también al departamento de la Universidad Politécnica de Cartagena de Ingeniería de Sistemas y Automática, a todo su personal y en especial al Dr. Jorge Feliú Batlle por su colaboración.

En tercer lugar, agradecer al grupo de investigación NEUROCOR, en especial a su investigador responsable D. Juan López Coronado por cederme sus instalaciones para la realización de este proyecto, y por su incansable apoyo y ánimo recibido.

Por últimO, y no menos importante, agradecer a todas aquellas personas que me han ayudado tanto personal como profesional y que es imposible mencionar a todas, ya que han sido numerosas las personas que de alguna forma u otra me han apoyado en este proyecto. De todas estas personas, agradecer en especial a la Ingeniera Industrial Doña Esther Conesa García, por su incansable apoyo y comprensión durante todos los años de formación, y en especial durante la realización de este proyecto.

Índice

Índice de figuras

Resumen y objetivos del Proyecto

Desde el departamento de Sistemas y Automática y en colaboración con el grupo de investigación NEUROCOR se propuso un proyecto basado en un exoesqueleto. El proyecto que se propuso se trataba de controlar las articulaciones de un prototipo dado. El proyecto se engloba dentro de un proyecto europeo llamado EXOLEGS.

El prototipo a controlar simula el tren inferior de un ser humano, este prototipo incluye las articulaciones de la cadera, rodillas y tobillos. Este proyecto está orientado a personas mayores y rehabilitación, por lo tanto, el prototipo tiene que ser capaz de realizar movimientos básicos para los seres humanos.

Se planteó la posibilidad de hacer el proyecto en Matlab debido a que éste se trata de un software matemático muy potente, y dispone de la herramienta de Simulink para realizar simulaciones dinámicas. El primer objetivo se trata de adaptar la librería del fabricante de los controladores a lenguaje Matlab ya que no se dispone de comunicación con dicho software. Una vez modificada la librería y conseguida una comunicación es el momento de realizar las rutinas de programación necesarias y óptimas para el control de cada una de las articulaciones. Todo el protocolo de comunicación se va a realizar mediante USB.

En segundo lugar se tiene que realizar una comunicación completa en la que se habilite el controlador. Para poder habilitar el controlador se necesita configurar cada uno de sus parámetros, así como el modo de funcionamiento, sensores, motores... Para toda esta parametrización se tiene que realizar una o varias clases con todas las propiedades necesarias, de esta forma se podrá modificar todo de una forma más sencilla y eficaz.

Para realizar el control del exoesqueleto se tienen que estudiar cada uno de los modos de operación disponible posibles y elegir el que más se adecue a el criterio establecido. Una vez elegido el modo de operación, para el control, se va a usar el controlador interno PID que dispone el controlador. La sintonización de este PID se llevará a acabo de forma teórica y experimental en vacío, y se comprobará con carga.

Para una simulación completa se necesita realizar un diagrama de bloques en Simulink. Para la simulación de las articulaciones se va a proporcionar una entrada de posición en grados. El diagrama de bloques tiene que estar optimizado para que se puedan añadir y quitar motores

de una forma fácil y rápida, sin necesidad de reprogramar todo el código. Para el análisis de resultados se dispondrá de una salida en tiempo real que comparará la posición objetivo con la posición real frente a un tiempo. En el diagrama de bloques se ha de ser capaz de poder modificar parámetros de los motores, tipos de motor e identificadores de cada controlador.

Para el proyecto se va a usar el prototipo inicial del proyecto EXOLEGS, esto no quiere decir que esta configuración sea la definitiva, por tanto, como objetivo principal de este proyecto, se plantea una rutina de comunicación en Simulink capaz de realizar labores de control y análisis de la respuesta del exoesqueleto ante diferentes entradas. Este modelo tiene que ser capaz de mediante parámetros modificar los diferentes cambios que se puedan realizar en el prototipo del exoesqueleto final, relación de transmisión, cambios de motor, cambios de identificadores...

En resumen, los objetivos del proyecto serían los siguientes:

1. Compatibilidad de librería del fabricante con lenguaje Matlab
2. Conexión y comunicación con la controladora
3. Sintonización del PID
4. Modelo en Simulink óptimo
5. Simulación en tiempo real
6. Problemas y mejoras

El proyecto se realizará en su totalidad en la Universidad Politécnica de Cartagena, en el laboratorio del grupo de investigación NEUROCOR, en este laboratorio se dispondrá de todo lo necesario para la realización del proyecto: CPU, controladores, prototipo, manuales del fabricante...

1. Introducción

1.1. Exoesqueletos

Los exoesqueletos tienen el objetivo de maximizar las habilidades físicas del ser humano, dotándolo de una gran fuerza, capaz de levantar hasta 10 veces su peso. El concepto de los exoesqueletos viene desarrollándose desde hace más de 30 años, en sus inicios fueron diseñados para uso industrial y con el paso del tiempo fueron evolucionando hasta llegar a lo que se conoce actualmente. A pesar de que muchos de estos trajes robóticos son diseñados para uso militar, también existen empresas dedicadas a la fabricación de exoesqueletos aplicados a la rehabilitación. En el siguiente artículo se hará un pequeño recorrido histórico y se conocerá alguno de estos trajes, desde el primer prototipo hasta algunos de los más avanzados.

1.1.1. Aplicaciones de exoesqueletos

Las aplicaciones de los exoesqueletos para personas pueden dividirse en tres importantes grupos:

- Exoesqueletos para rehabilitación de personas que han sufrido accidentes o padecen enfermedades que pueden mejorar los síntomas con ejercicios controlados.
- Exoesqueletos para hacer que una persona en buen estado físico pueda realizar un mayor esfuerzo (por ejemplo, exoesqueletos militares o prototipos ideados para que un trabajador pueda portar pesos mayores durante la jornada laboral).
- Exoesqueletos para ayudar a personas a llevar una vida menos dependiente. En este caso están las personas con movilidad reducida, ya sea por accidentes que les han quitado la movilidad de sus extremidades, por enfermedades crónicas o por los efectos de la edad, ancianos que ven reducidas sus capacidades físicas.

A pesar de que durante el uso de exoesqueletos con los fines del primer grupo, rehabilitación, y del último, servir de proporcionar movilidad a personas, se realiza un ejercicio que puede mejorar la movilidad y la forma física del usuario, hay que realizar una separación entre ambas. Esta separación se ha hecho teniendo en cuenta que el primer grupo son situaciones transitorias con un tiempo relativamente corto del uso del exoesqueleto (con un horario diario o semanal fijado hasta que la persona es capaz de volver a la normalidad tras la rehabilitación), y el último grupo se ha considerado un uso cotidiano (sin horario fijo, estando el exoesqueleto a disposición de la persona para su uso privado) debido al carácter crónico de la enfermedad o debido al estado y edad de la persona.

En este caso, el proyecto EXOLEGS está centrado en exoesqueletos para personas ancianas, con baja movilidad y que la ayuda de un exoesqueleto les ayudaría a llevar una vida lo menos dependiente posible.

1.1.2. Ejercicio físico en personas mayores

Desde el nacimiento hasta las últimas horas de vida, es a través del cuerpo y del movimiento la manera mediante la cual interaccionamos constantemente con nuestro entorno. Cuerpo y movimiento son dos términos estrechamente relacionados entre sí. El movimiento garantiza el correcto funcionamiento del organismo y a su vez, un cuerpo sano es base fundamental para el movimiento libre. La realización de actividad física sistemática y controlada es una clara herramienta de influencia positiva para preservar, conservar y promocionar los múltiples factores que intervienen en el mantenimiento de un cuerpo sano a lo largo de los años.

Se aconseja la práctica regular de ejercicio físico en mayores de 65 años, dado que esto conlleva efectos beneficiosos sobre la diabetes, la hipertensión, las caídas, el nivel de independencia, los niveles de colesterol y la enfermedad coronaria, entre otras patologías.

Un exoesqueleto puede proporcionar a determinadas personas parte de esa movilidad extra que necesitan para salir de ese anquilosamiento muscular y físico, reactivar el movimiento, caminar más a menudo (realizando un menor esfuerzo) realizar tareas cotidianas simples. De esta forma realizar estas tareas cotidianas es realizar ejercicios leves que conllevan a una mejora en la condición física de la persona. Los movimientos realizados en las extremidades inferiores con ayuda del exoesqueleto y los realizados en las extremidades superiores debido a que ahora si pueden desplazarse con facilidad para realizar tareas, así como la actividad en el tronco de la persona son ejercicios que antes de usar el exoesqueleto no se realizaban porque eran muy costosos, o si se realizaban podrían dar lugar a lesiones, y ahora ayudan a una mejora continua en la persona.

Ventajas del uso de exoesqueletos en personas mayores.

Ya se han comentado en el párrafo anterior muchas de las ventajas que tiene el uso de exoesqueletos, así, se pueden resumir en los siguientes puntos:

- Previene la atrofia de los músculos y puede iniciar un proceso de recuperación en caso de que se haya producido esa atrofia.
- Aumenta la autonomía de la persona.

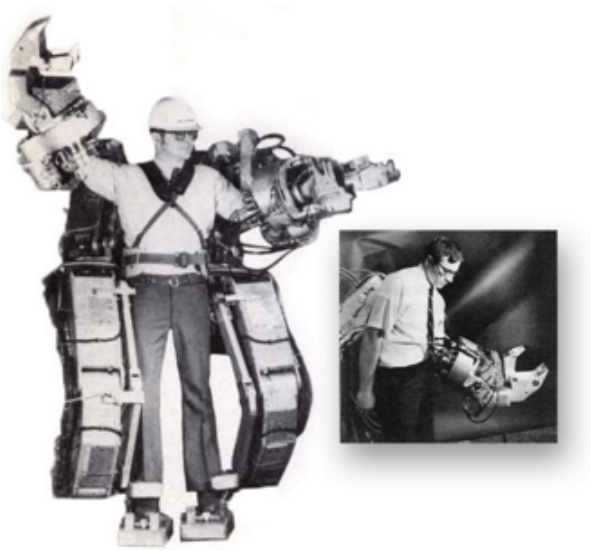
- La menor dependencia y el ejercicio físico crea una seguridad en la persona que le ayuda a mejorar su estado psicológico.
- No es necesario adaptar el entorno de la persona que porta un exoesqueleto.
- Reduce el esfuerzo que necesita la persona para moverse.
- El usuario no pone todo su peso en las articulaciones, produciendo un desgaste menor.
- Se puede hacer estudios centrados en articulaciones o músculos concretos.

1.1.3. Evolución de los exoesqueletos

Tipos de exoesqueletos hay diversos, ya que se puede robotizar cualquier parte del cuerpo humano de forma mecánica, pero este proyecto tiene como interés el tren inferior del cuerpo humano. En el tren inferior se encuentran la cadera, las rodillas y los tobillos. Para realizar una evolución de los exoesqueletos nos centraremos en los que mas se asemejan a este proyecto, es decir, aquellos que están enfocados al tren inferior humano, y orientados a la rehabilitación.

Proyecto Hardiman

En la década de los 60, Ralph Mosher, ingeniero de General Electric diseñó un traje robótico capaz de potenciar la fuerza humana, brindándole al usuario la posibilidad de cargar hasta 1500 kilogramos sin mayor dificultad. Hardiman fue el primer proyecto serio de un exoesqueleto robótico y el más ambicioso de General Electric y digno de admiración para la época en que fue creado. General Electric tenía grandes expectativas para el proyecto Hardiman, pues esperaba que fuera utilizado a bordo de portaaviones para la carga de bombas, construcción submarina, en centrales nucleares, y en el espacio exterior. Lamentablemente el proyecto fracasó, la incontrolable violencia de sus movimientos la ineficiencia de sus piernas que componían la base del traje no permitieron que llegara a tener un desarrollo óptimo. Al fracasar el proyecto Hardiman decidieron reducir medidas y crearon un brazo robótico. Esta especie de guante biónico era capaz de levantar hasta 340 kilogramos de peso. Irónicamente, este guante biónico pesaba el doble de lo que podía levantar, 680 kilos. No, no era muy útil, así que la investigación fue cancelada.



Lifesuit

En 1986 Monty Reed, comenzó a trabajar en un traje robótico que puediera ser utilizado como herramienta de rehabilitación. El traje pesaba unas 75 libras, era una combinación de una mochila y un paquete de cohetes y tanques de buceo. En los últimos 20 años Monty Reed ha destinado su propio dinero al desarrollo e investigación de un exoesqueleto capaz de devolverle la habilidad de caminar incluso a personas con cuadriplejia. Hasta el momento se han creado varias versiones del Lifesuit obteniendo muy buenos resultados, actualmente el LIFESUIT 14 tienen la capacidad de caminar una milla llevando un peso extra de 92 kilogramos.

Traje de apoyo asistido

En 1990 los científicos del Instituto de Tecnología de Kanagawa d en Japón diseñaron un traje robótico de asistencia, el cual estaba destinado a simplificar la labor de las enfermeras. Este traje constaba con extremidades plegables, las cuales funcionaban con tensores de aire comprimido que reaccionaban gracias a información enviada por un ordenador que controlaban los sensores de esfuerzo del usuario.Si la enfermera decidía movilizar a un paciente, el traje se activaba facilitando el traslado del enfermo, minimizando el esfuerzo de las zonas activas de la enfermera como el tronco, hombros y cintura. Una enfermera de 67 kilogramos podía levantar a un paciente de 70 kilogramos sin esfuerzo. Este traje fue diseñado con la finalidad de evitar las lesiones de espalda que muchas veces sufre el personal medico al tratar pacientes con movilidad limitada.

HAL de Cyberdyne

Desde el año 2002 Yoshiyuki Sankai, investigador de robótica de la Universidad de Tsukuba en Japón y fundador de la empresa japonesa Cyberdyne, ha venido trabajando en un traje robótico diseñado para ayudar a ancianos y personas discapacidades motoras. EL traje HAL (Hybrid Assisting Limb), es un robot cibernético con elementos de la biónica, robótica y electrónica. Después de varias versiones actualmente es un traje de solo 25 kilos es capaz de multiplicar la fuerza del usuario de dos a diez veces.En el 2009 la empresa Cyberdyne, anunció la fabricación en masa del HAL para ser comercializado, ofreciendo dos opciones, el traje completo y otro solo para miembros inferiores.

1.1.4. Exoesqueletos en la actualidad

En la actualidad existen diversos tipos de exoesqueletos, desde aquellos que simulan un cuerpo humano al completo hasta aquellos que realizan una única acción. Como objeto de este proyecto se van a hablar de los exoesqueletos actuales orientados al tren inferior humano, y más concretamente a personas mayores y a rehabilitación. Actualmente los proyectos más desarrollados y comercializados los tienen, entre otras, las siguientes empresas, los cuales detallaremos a continuación: Ekso Bionics, Rex Bionics y Cyberdine.

Ekso Bionics

EKSO es un traje biónico portátil que permite a las personas con limitaciones en las extremidades inferiores ponerse de pie y caminar por un terreno natural, proporcionan un apoyo completo al suelo. La acción de caminar es posible cuando los sensores del exoesqueleto detectan peso, entonces se inicia la acción de caminar. Todo el peso y las acciones de las extremidades deficientes son realizadas por motores alimentados con baterias. Este exoesqueleto proporciona la posibilidad de ponerse de pie y caminar para personas con discapacidad total. Ayuda a los pacientes volver a aprender los patrones de pasos adecuados.

EKSO es un exoesqueleto para su uso bajo supervisión médica por individuos con diferentes niveles de parálisis o hemiparesia debida a enfermedades neurológicas como el ictus, lesión de la médula espinal o enfermedad, lesión cerebral traumática y más. Con el alta médica, por lo general facilita el caminar de las personas con una amplia gama de habilidades motoras y tamaños. Todo el mundo médicamente tratado que ha pasado el examen físico ha caminado en su primera sesiónDiseñado para la utilidad y facilidad de uso en un entorno clínico

Rex Bionics

REX es un dispositivo de movilidad robótica manos libres para la rehabilitación. Diseñado para las personas con problemas de movilidad, REX es completamente autosuficiente y rápidamente ajustable para cada usuario, se abre una gran posibilidad de rehabilitación para una amplia gama de personas.Rex Bionics está trabajando con los fisioterapeutas para desarrollar la práctica de la fisioterapia asistida por robot (RAP). En una sesión de RAP, REX levanta a los pacientes de una posición sentada a una posición de pie, lo que les permite caminar de forma segura y poder hacer ejercicios de estiramiento, diseñados por fisioterapeutas especializados.

REX también limita la carga sobre los terapeutas durante la manipulación manual y reduce la cantidad de ayuda necesaria para ayudar a los pacientes que se someten a terapia de pie.Las personas en silla de ruedas se encuentran en riesgo de padecer numerosas complicaciones médicas por largos periodos de estar sentado, por lo que este dispositivo les permite pasar más tiempo de pie, caminar y hacer ejercicio. REX puede ofrecer beneficios significativos para la salud. Un programa de ensayos clínicos está actualmente en curso para evaluar estos beneficios potenciales para la salud de las personas.

Cyberdyne

El modelo de Cyberdine se llama HAL. Las principales causas de las discapacidades de las extremidades inferiores son trastornos del sistema muscular y nervioso cerebral. En esos casos, el cerebro no puede utilizar las vías neurales ordinarios y no puede ordenar de mover las piernas. HAL para uso médico mueve las piernas del usuario de acuerdo con la intención del usuario, "Quiero caminar..º "Quiero estar de pie.", Y permite la retroalimentación oportuna de los sentimientos ", que podía caminar. .º"Yo podría ponerse de pie. ". En consecuencia, esto acelera el aprendizaje por el cerebro. HAL® para uso médico es el único dispositivo de recuperación robótico que puede enseñar el cerebro cómo mover las piernas. Este dispositivo se puede aplicar a personas que sufren de lesiones de la médula espinal, lesiones cerebrales traumáticas, enfermedades cerebrovasculares, enfermedades del cerebro y del sistema neuromuscular, etc.

Cuando una persona se mueve, el cuerpo envía diferentes señales desde el cerebro a los músculos a través de nervios. Esas señales aparecen en la superficie de la piel como "señales bio-eléctrica [BES]". HAL lee el Modelo BES del usuario, por lo tanto compensa la fuerza muscular de las extremidades inferiores y él o ella ayuda a caminar, pararse, y sentarse de forma autónoma.

(a) Ekso Bionics

(b) Rex Bionics

(c) HAL de Cyberdine

1.2. EXOLEGS

La mayoría de las personas de edad sufren de leve a grados agudos de la degeneración física y cognitiva. El carácter progresivo de estos trastornos a menudo conduce a la pérdida de la independencia que afecta la calidad de vida. Sin embargo, se ha demostrado que la movilidad física y el ejercicio cardiovascular mejora directamente las capacidades cognitivas humanas. Por lo tanto, esta propuesta va a desarrollar una gama de soluciones de asistencia activa exoesqueléticas de miembros inferiores (Basic, Standard y Deluxe) para proporcionar la movilidad interior y exterior para ayudar a las personas mayores a mantener y mejorar los niveles de actividad física para una vida plena, activa e independiente.

Los requisitos de movilidad demandados al exoesqueleto para que las personas de edad avanzada puedan llevar a cabo sus actividades cotidianas de una manera independiente son los siguientes:

- Movilidad interior: mover libremente dentro de espacios cerrados que dan un valor añadido considerable sobre sillas de ruedas, sea capaz de realizar maniobras de sentarse y levantarse, subir y bajar escaleras, pasar por encima de objetos, caminar recto.
- Movilidad al aire libre: caminar sobre diferente tipo de superficies y sobre superficies irregulares, evitar el tráfico cruzando las carreteras, poder acceder al transporte público, usar escaleras mecánicas.
- Apoyo cognitivo: proporcionar información para permitir la toma de decisiones cuando la persona mayor se encuentra perdida o en un estado de confunsión.

Objetivos

EXOLEGS tiene como finalidad el desarrollo de exoesqueletos de movilidad corporal del tren inferior para ayudar a las personas a realizar tareas normales de la vida diaria. Los movimientos incluyen tareas como levantarse, sentarse, caminar en línea recta en terreno plano, pasar por encima de los objetos, caminar sobre terreno blando y desigual, subir y bajar escaleras, etc. En el proyecto EXOLEGS, se desarrollarán marcos teóricos y modulares capaces de realizar prototipos de dispositivos que pueden ser útiles para ayudar a la movilidad humana. Existirán 3 tipos de exoesqueletos: Basic, Standard y Deluxe, que serán diseñados, desarrollados y probados en los siguientes países de la Unión Europea: Suecia, Alemania, España, Reino Unido y Suiza.

Valor Agregado sobre sillas de ruedas

El valor añadido de EXOLEG en función de las soluciones de movilidad en silla de ruedas actuales, es que va a proporcionar a los usuarios la capacidad de moverse en diferentes tipos de suelo (duro, blando, irregular, etc), subir y bajar escaleras, y pasar por encima de los objetos. Por otra parte, los exoesqueletos tienen un tamaño considerablemente menor que las sillas de ruedas a la hora de desenvolverse por una casa doméstica. Los exoesqueletos imitan la postura normal del cuerpo, su uso tendrá beneficios significativos para la salud, a diferencia de las sillas de ruedas, (como la recuperación de las funciones normales de vejiga e intestino), proporcionando así la capacidad de hacer ejercicio y aumentar la actividad muscular. Es bien sabido que tal ejercicio aumenta las habilidades cognitivas a medida que aumenta el flujo de oxígeno al cerebro, dando otra razón para elegir esta opción sobre las sillas de ruedas.

La clave del éxito de adopción de las soluciones EXOLEGS se basan en los sistemas de seguridad (cumpliendo con existente y las normas de seguridad ISO / IEC emergentes), inspira confianza en el usuario y siendo accesible a través de un modelo de negocio adecuado (alquiler, arrendamiento, uso de varias personas, prescripción, etc.). El proyecto EXOLEGS tendrá en cuenta todas estas cuestiones en el plan de trabajo detallado.

1.2.1. Prototipo

Para conseguir el objetivo de este proyecto se cuenta con un prototipo previamente fabricado. El prototipo ha sido diseñado y dimensionado en España. Este prototipo cuenta con el tren inferior completo de una persona humana. Tanto el hadware de control como el prototipo ha sido construido y escogido previamente a este proyecto.

El prototipo consta principalmente de dos partes: la estructura y los actuadores y sensores. En referencia a la estructura se puede observar que es una estructura metálica, aparentemente no parece muy pesada pero en la práctica cuenta con un peso aproximado de 30Kg. Para la sujeción con la persona cuenta con bandas de tela tanto en la cadera como a lo largo de toda la pierna. Para realizar todo este proyecto el prototipo ha estado unido en la parte inferior por una plancha de acero.

En la parte referente a los actuadores y sensores se observa que hay 6 motores. Cada motor cuenta de forma integrada con un encoder y una reductora, y de forma externa y de unión con las extremidades con un engranaje cónico reductor. Todos los sensores integrados son los mismos para todos los motores. En la zona de la cadera se encuentra un motor de 70W de la marca Maxon, y para la zona de las articulaciones de la rodilla y tobillo, se encuentran unos motores de 100W de la marca Maxon.

Figura 1: Prototipo.

1.2.2. Motores

En el prototipo que se ha proporcionado tiene instalado dos tipos de motores. En primer lugar, para la cadera, se usa un motor EC45 (70W) con referencia nº397172 del catálogo Maxon, y para las rodillas y tobillos, se ha utilizado un motor EC60 (100W) con referencia nº412825 del catálogo Maxon. A continuación se detallarán los parámetros usados de los motores.

EC45 (70W)

- MotorType = 10
- NominalCurrent = 3210
- MaxOutputCurrent = 6420
- ThermalTimeConstant = 29.6
- NbOfPolePairs = 8

EC60 (100W)

- MotorType = 10
- NominalCurrent = 3210
- MaxOutputCurrent = 6420
- ThermalTimeConstant = 40
- NbOfPolePairs = 7

1.2.3. Sensores

En la parte de sensores, en este prototipo se tienen instalado 6 encoders, 1 en cada motor. En este caso, nos encontramos con un encoder rotatorio, el cual se encarga de decirnos en que posición está el motor en cada momento. Los encoders que llevan montados estos motores son incrementales, y tienen un problema asociado a este proyecto desde un punto de vista de seguridad. Estos sensores se reinician en el momento que se le quita la alimentación, por tanto cuando se alimentan por primera vez después de desconectarlos es imposible saber en que posición se encuentra la estructura sin verla. Los encoders usados tienen como referencia nº421986 del catálogo Maxon, y ahí se pueden ver todas sus características. A la hora de configuración y operación sus parámetros son los siguientes:

- SensorType = 2
- EncoderResolution = 1024

1.2.4. Engranajes

Cada uno de los motores, aparte del sensor, lleva integradas unas reductoras, las cuales se encargan de transmitir el par y de hacer más sencillo su control y precisión reduciendo la velocidad.

En el interior de cada motor se encuentran unas reductoras planetarias de acero inoxidable. En cada articulación se tendrán diferentes relaciones de transmisión en función de las necesidades. En la articulación de la cadera se encontrará una reductora planetaria 53:1 con referencia nº203121 en el catálogo Maxon. En las articulaciones de las rodillas y tobillos, se tendrán las mismas configuraciónes de reductoras, por lo tanto, en ambas se encontrarán unas reductoras planetarias 19:1 con referencia nº223085 del catálogo Maxon.

Como unión de los actuadores con las articulaciones, se encuentran unos engranajes cónicos helicoidales integrados en la estructura. Estos engranajes se tratan como una reductora 3:1 la cual se encarga de transmitir el par de la reductora planetaria de cada motor y de reducir la velocidad aún más.

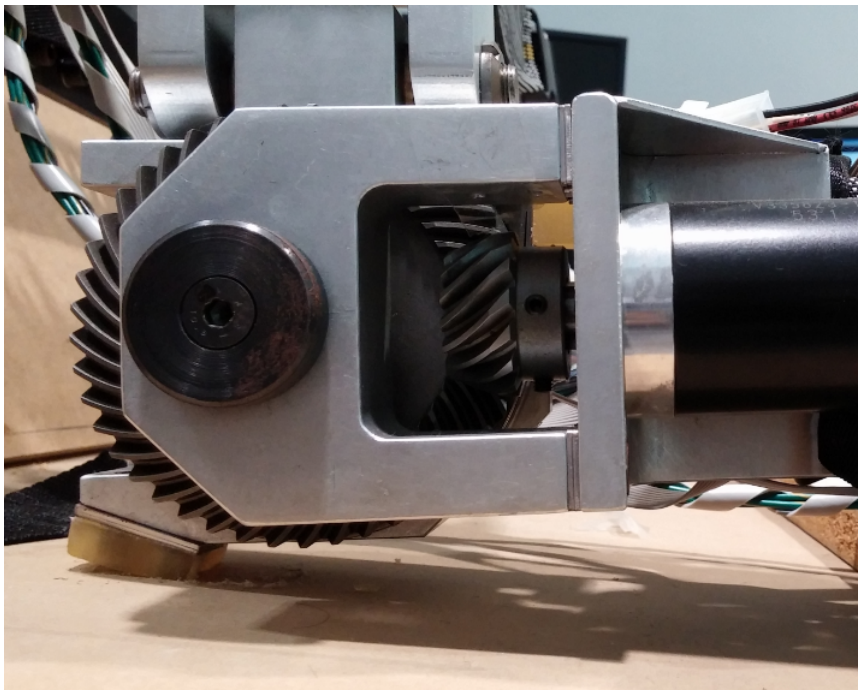


Figura 2: Engranaje cónico helicoidal 3:1.

1.3. Hadware

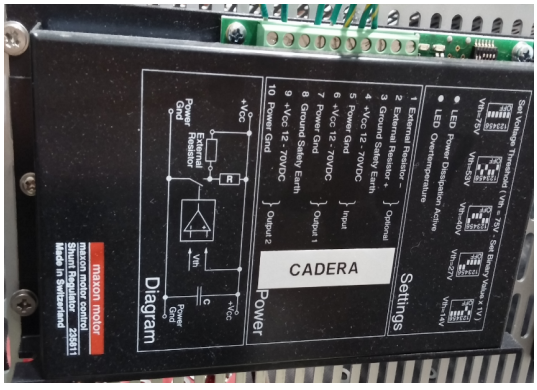
Para la realización del proyecto se ha necesitado un hadware mínimo: un ordenador o CPU, un derivador de voltaje como medida de protección ante sobretensiones, una fuente de alimentación o diferentes fuentes de alimentación capaces de proporcional la potencia necesaria y por último las controladoras.

1.3.1. CPU

Como CPU y para realizar los ensayos, se ha utilizado un ordenador de sobremesa con Windows 7, el cual es capaz de mover el software necesario para el control en tiempo real del prototipo. En el proyecto real este ordenador no es un sobremesa, se trata de un miniordenador con un sistema operativo basado en software libre (linux).

1.3.2. Regulador de voltaje

Como medida de protección ante sobretensiones se tienen 3 reguladores de voltaje. Estos reguladores de voltaje se configuran para suministrar un voltaje máximo a su salida, y si hay sobretensiones éstos las eliminan en forma de calor. Para este proyecto se van a usar 3 reguladores ya que cada regulador es capaz de suministrar la potencia necesaria para dos controladoras con sus respectivos actuadores. Para la configuración del regulador de voltaje y el voltaje a la salida de éste, se dispone de unos microinterruptores los cuales estan compuestos por 6 microinterruptores que son capaces de establecer el límite de voltaje a la salida del regulador. Los reguladores son de la marca Maxon.



(a) Derivador de voltaje

1.3.3. Fuente de alimentación

Para la alimentación de las controladoras y los motores se usan dos fuentes de alimentación diferentes. Para los motores de menor potencia que son los de la cadera (EC45) se dispone de una fuente para ellos que suministrará 24V y 2A. Para la articulación de las rodillas y tobillos se dispone de una fuente capaz de suministrar más potencia ya que cada motor tiene una potencia de 100W.

(b) Fuente Cadera

(c) Fuente Rodilla y Tobillo

1.3.4. Controlador

La parte o el hadware más importante para el proyecto es el controlador, que es la que se encarga de suministrar la potencia necesaria en cada momento a los motores para moverlos. El controlador se trata de un hadware de la marca Maxon y es el modelo EPOS2 50/5. EPOS2 50/5 es un controlador digital de posición modular. Apto para motores CC de escobillas con encoder o para motores EC sin escobillas con sensores Hall y encoder desde 5 hasta 250 W. Este controlador puede funcionar en diferentes interfaces: USB, RS232 y CAN. En el interior del controlador tiene un controlador integrado, también es capaz de regular la corriente, la velocidad y la posición de los motores.

(d) Controlador EPOS2 50/5

1.4. Software

Para la realización de este proyecto se han usado dos software diferentes. En primer lugar, para el testeo y correcta configuración de las controladoras y motores, se ha usado un software proporcionado por el fabricante, Epos Studio. Este software permite conectar y trabajar de manera inmediata con los motores. En segundo lugar, para la realización del proyecto, obtención de datos, control y operación, se ha usado el lenguaje Matlab en el entorno de programación de Matlab. Para realizar pruebas a tiempo real se ha optado por el módulo de Matlab que se llama Simulink, el cual se trata de un entorno de programación mediante bloques que permite simulaciones dinámicas

1.4.1. Epos Studio

Epos Studio es un software desarrollado por el fabricante de los controladores, motores, sensores y engranajes del exoesqueleto. Este software lo proporciona la empresa Maxon con el fin de poder trabajar de forma sencilla con los motores. Este software permite analizar cualquier parámetro de configuración de los controladores, motores y sensores en cualquier instante de tiempo, siempre y cuando estén conectadas.

En primer lugar, para la utilización de este software es necesario tener conectadas y correctamente alimentados los controladores, motores y sensores. En este proyecto solo se ha usado este software para testear el funcionamiento de los motores y para adquirir parámetros de configuración que no venían dados en ninguna guía.

Este software permite conectar todas los controladores a la vez y mediante diferentes protocolos de comunicación, el software se adapta al protocolo de comunicación que hayamos elegido. Una vez creado el proyecto, el software muestra un esquema de conexión donde pone los identificadores y puertos de conexión de cada uno de los controladores, de esta forma se sabrá en que puerto esta conectado cada uno y el identificador de nodo que tenga cada controlador si se trabaja con el protocolo de comunicación CANOpen.

Para poder realizar correctamente el testeo del montaje en cuestión, se puede hacer con cualquier modo de operación que se estime oportuno. En los diferentes modos de operación se pueden modificar diferentes parámetros. Por último este software tiene una aplicación que permite sintonizar de manera automática el controlador PID de la controladora más adecuado.

1.4.2. Matlab

Matlab es una herramienta de software matemático que ofrece un entorno de desarrollo integrado con un lenguaje de programación, a este lenguaje se le denomina lenguaje Matlab o M. Entre sus prestaciones básicas se hallan: la manipulación de matrices, la representación de datos y funciones, la implementación de algoritmos, la creación de interfaces de usuario y la comunicación con programas en otros lenguajes y con otros dispositivos hardware.

El paquete Matlab dispone de dos herramientas adicionales que expanden sus prestaciones, Simulink y Guide.Simulink permite una programación basada en bloques para simulaciones dinámicas, y Guide permite la creación de interfaces de usuario de manera sencilla. Además, se pueden ampliar las capacidades de Matlab con las cajas de herramientas (toolboxes), y las de Simulink con los paquetes de bloques (blocksets).

Las aplicaciones de Matlab se desarrollan en un lenguaje de programación propio. Este lenguaje es interpretado, y puede ejecutarse tanto en el entorno interactvo, a través de un archivo de script (archivos *.m). Este lenguaje permite operaciones de vectores y matrices, funciones, cálculo lambda, y programación orientada a objetos.

Simulink es un entorno de diagramas de bloque para la simulación multidominio y el diseño basado en modelos. Admite el diseño y la simulación a nivel de sistema, la generación automática de código y la prueba y verificación continuas de los sistemas embebidos. Simulink ofrece un editor gráfico, bibliotecas de bloques personalizables y solvers para modelar y simular sistemas dinámicos. Se integra con Matlab, lo que permite incorporar algoritmos de Matlab en los modelos y exportar los resultados de la simulación a Matlab para llevar a cabo más análisis. Es de destacar que no toda la funcionalidad del lenguaje Matlab se puede extrapolar a la programación en Simulink debido a algunas incompatibilidades con funciones prediseñadas en Matlab, un ejemplo de ello es la función delay.

2. Desarrollo de Algorítmo de comunicación con Hadware

2.1. Comunicación

En este apartado, se va a desarrollar la rutina creada para una comunicación exitosa entre el controlador y el software de control. Para realizar la conexión, se va a necesitar una biblioteca específica para aprovechar al máximo nuestro hadware. Se presentó un primer problema, y fue que no había unos drivers específicos para Matlab, nuestro software de control. Una solución fue contactar con una persona que modificó los drivers para adaptarlos a Matlab. Una vez que se ha cargado la librería satisfactoriamente, se podrá empezar a comunicar con el hadware, pero antes de utilizarlo hay que configurarlo, ya que este hadware tiene diferentes parámetros para según el motor y sensores a los que esté conectado.

2.1.1. Librerias y configuración necesaria

Como se ha dicho anteriormente, lo necesario para tener una comunicación, es disponer de una librería que sea capaz de poder conectar el software de control con el hadware. Se va a usar una librería modificada para Matlab, disponible tanto para versiones de 32bit como de 64bit. Esta librería se denomina EposCmd.dll para 32bit y EposCmd64.dll para 64 bit. No solo se necesita una librería para poder conectarlo, también se necesita un archivo denominado Definitions.h, el cual se encarga de definir cada una de las funciones de la librería, así como cada uno de los tipos de datos, constantes, etc.

El archivo Definitions.h se usa para poder cargar la librería. Este archivo se encarga de compatibilizar las funciones de la librería y definir los tipos de datos que se están usando. Este archivo se tuvo que modificar debido a una incompatibilidad en el tipo de dato. Se modificó un tipo de dato que no se reconocía, por uno equivalente reconocido por nuestro software. Se definieron estos datos nuevamente y se solucionó el error de comunicación.

A la hora de cargar la librería en Matlab mostraba una serie de errores, los cuales no permitían cargar la librería. Eran numerosos errores, pero analizando, se observo que se trataba en todos los casos del mismo tipo de error. El error consta de una incompatibilidad de matlab con el tipo de dato __ int8 e __ int8Ptr. Estos tipos de datos corresponden en su equivalente matlab al tipo de dato int8 e int8* . A continuación se muestra la solución propuesta, la cual funciona de forma correcta y permite cargar la librería sin errores.

```
#define __int8 int8;
#define __int8Ptr int8*;
```

Con esta modificación, se tiene una librería compatible con Matlab, que servirá para comunicarse con la controladora.

Una vez se ha cargado la librería, se tendrá que configurar el controlador para nuestros elementos de sensorización y motores. Para que el control sea óptimo se tendrá que parametrizar y configurar el controlador de la siguiente manera, y con el siguiente orden:

- Establecer diálogo con el hadware
- Establecer un protocolo de comunicación
- Limpiar los errores guardados
- Configuración de parámetros de operación
- Configuración de Motores y Sensores
- Habilitar controlador

Todas estas rutinas, se desarrollaran más adelante, en un apartado específico para la rutina de conexión. Si toda esta configuración se ha llevado a cabo sin ningún error, se tendrá el controlador conectada al software, con el cual se podrá controlar los motores y recibir la posición de las articulaciones en cada instante de tiempo. Para concluir con la comunicación, se tendrá que realizar una desconexión exitosa, para no tener problemas de conexión en un futuro. Para realizar la desconexión, se realizará en el siguiente orden.

- Deshabilitar controlador
- Desconectar dispositivos
- Quitar librería de memoria

2.2. Modos de operación

El controlador dispone de diferentes modo de operación, con los cuales se puede establecer un modo de trabajo del motor que mas se adapte a el objetivo. Existen varios modos de operación, cada uno tiene unos parametros prioritarios de funcionamiento, por lo tanto, se tendrá una variedad de modos de operación para escoger el que más se ajuste. A continuación se desarrolla en una tabla los diferentes modos de operación, y sus valores para seleccionarlos en el controlador cuando se está configurando.

Description	Value	Name
Position Profile Mode	1	OMD_PROFILE_POSITION_MODE
Position Velocity Mode	3	OMD_PROFILE_VELOCITY_MODE
Homing Mode	6	OMD_HOMING_MODE
Interpolated Position Mode	7	OMD_INTERPOLATED_POSITION_MODE
Position Mode	-1	OMD_POSITION_MODE
Velocity Mode	-2	OMD_VELOCITY_MODE
Current Mode	-3	OMD_CURRENT_MODE
Master Encoder Mode	-5	OMD_MASTER_ENCODER_MODE
Step Direction Mode	-6	OMD_STEP_DIRECTION_MODE

Figura 3: Modos de operación.

Profile Position Mode

En el modo de operación de perfil de posición, se tiene que el objetivo principal es la posición, es decir, el número de pasos que se pretende que el motor avance, o en el caso de el exoesqueleto, el número de grados que se quiere girar las articulaciones. La característica de este perfil, es la capacidad de realizar ese movimiento mediante diferentes trayectorias, trapezoidal y senoidal.

Como se dispone de estos dos tipos de perfil para generar la trayectoria, se tendrá que decidir cual es el mejor para cada uno de los movimientos. Para realizar unos movimientos lo más natural posible, necesitaremos un perfil de velocidad acampanado, de forma que no acelere de forma lineal, y que cuando vaya a llegar a la posición vaya cada vez más lento. En función de estas premisas, y analizando el comportamiento de los perfiles Trapezoidal y Senoidal, se elegirá para los movimientos un perfil Senoidal.

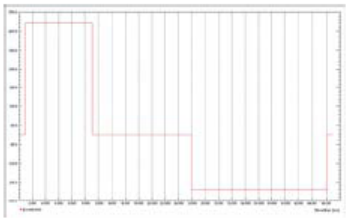
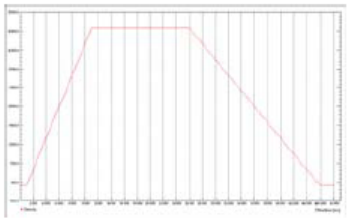
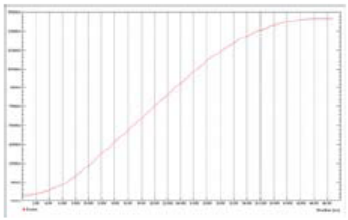
Acceleration	Velocity	Position
		
A_{max} = Profile acceleration A_{min} = Profile deceleration	V_{max} = Profile velocity	Absolute Movement P_{end} = Target position Relative Movement P_{end} = Position demand value + Target position

Figura 4: Perfil de posición, Trapezoidal.

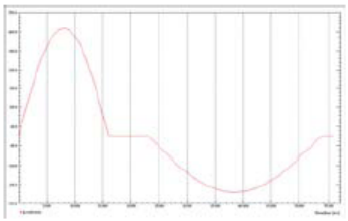
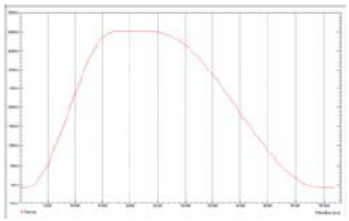
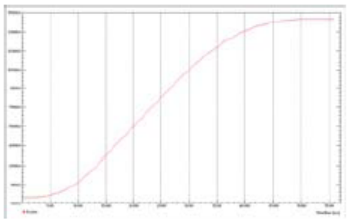
Acceleration	Velocity	Position
		
A_{max} = Profile acceleration A_{min} = Profile deceleration	V_{max} = Profile velocity	Absolute Movement P_{end} = Target position Relative Movement P_{end} = Position demand value + Target position

Figura 5: Perfil de posición, Senoidal.

Se observa, que para establecer nuestro perfil de posición, se exige una serie de parámetros. Estos parámetros se podrán modificar en cualquier momento, sin necesidad de apagar la controladora. Los parámetros con los cuales se puede trabajar son:

- Perfil de Velocidad
- Perfil de Aceleración
- Perfil de Deceleración

Mediante el Perfil de Velocidad, se puede modificar la velocidad con la que se quiere mover la articulación, medidas en rpm. Mediante el perfil de Aceleración y Deceleración, se establecen los parámetros para generar dichas trayectorias, limitando dichas trayectorias a los valores establecidos.

Profile Velocity Mode

En el modo de operación de perfil de Velocidad, se trabaja configurando una trayectoria de velocidad, junto con una función de control de velocidad. El objetivo de este modo de operación, a diferencia del de perfil de posición, es una velocidad. Igual que en el modo de perfil de Posición, se tendrá una trayectoria Trapezoidal y una trayectoria Senoidal.

Se puede observar en las siguientes figuras de los perfiles, que el perfil que mejor se adapta a unos movimientos naturales de las articulaciones, sigue siendo el perfil Senoidal.

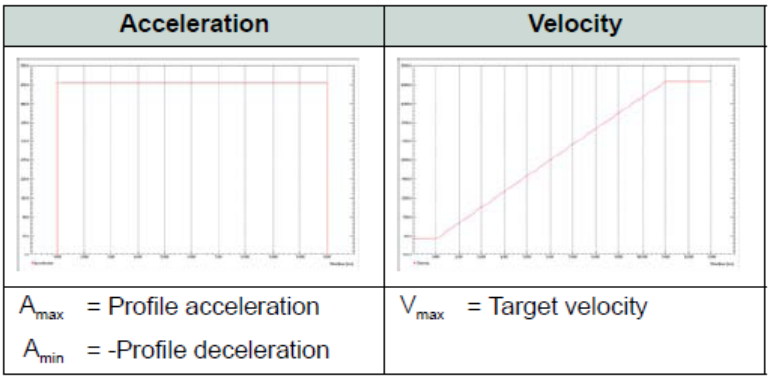


Figura 6: Perfil de velocidad, Trapezoidal.

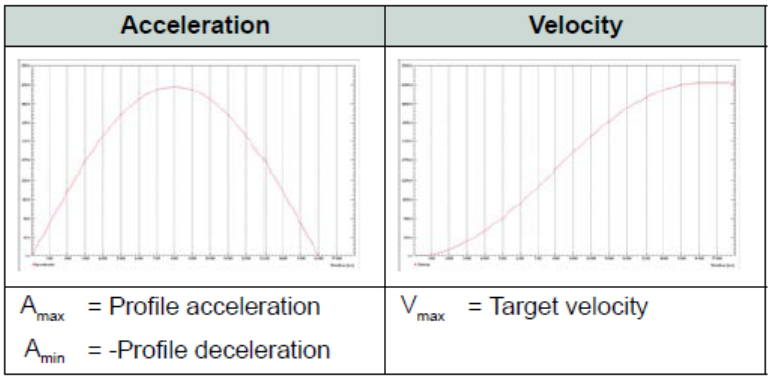


Figura 7: Perfil de velocidad, Senoidal.

Para la configuración de las trayectorias de velocidad, se exige una serie de parámetros. Estos parámetros se podrán modificar en cualquier momento, sin necesidad de apagar la controladora. Los parámetros con los cuales se puede trabajar son:

- Perfil de Aceleración
- Perfil de Deceleración

Mediante el perfil de Aceleración y Deceleración, se establecen los parámetros para generar dichas trayectorias, limitando dichas trayectorias a los valores establecidos.

Homing Mode

El modo de operación Homing, se usa para encontrar la posición de referencia o posición cero. Para esto hay diferentes formas, pero una de las principales se trata, con un encoder incremental, de buscar la posición cero, y de esta forma, se obtendrá la posición de referencia, otro método, consiste en la utilización de microinterruptores.

En este modo de operación, también se generan trayectorias, y por lo tanto, tendremos trayectorias Trapezoidal y Senoidal. Se elegirá siempre una trayectoria Senoidal, debido a su naturalidad a la hora de realizar movimientos.

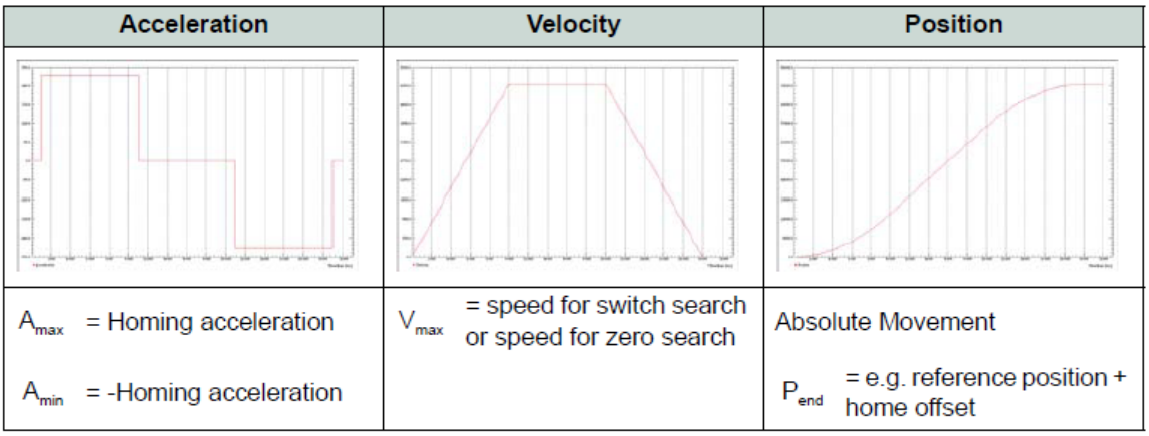


Figura 8: Homing Mode, Trapezoidal.

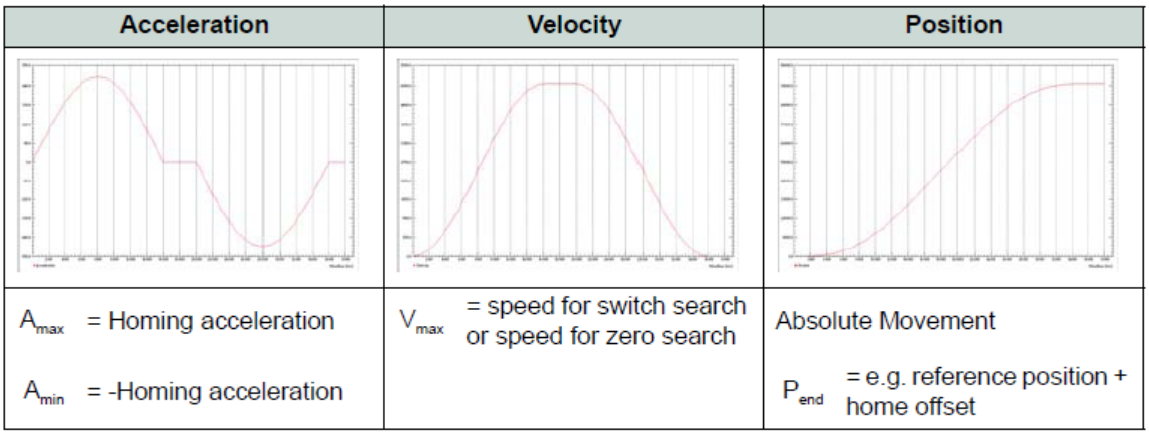


Figura 9: Homing Mode, Senoidal.

Interpolated Position Mode

El modo de operación de Interpolación, funciona como su propio nombre indica, con un método que interpola dos puntos dados, los cuales proporcionan información de posición, velocidad y tiempo. Con estos datos se genera una trayctoria para un tercer punto, a continuación se presenta una imagen como ejemplo de funcionamiento.

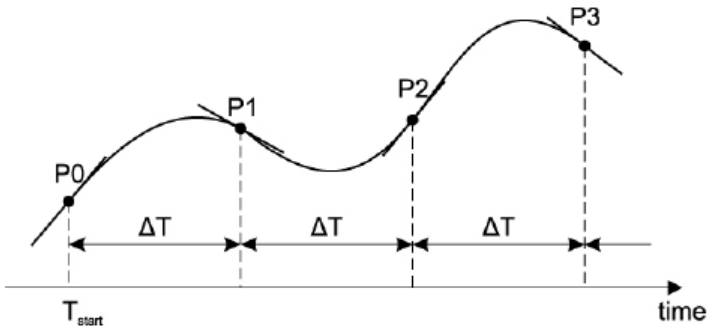


Figura 10: Interpolated Position Mode, Ejemplo de interpolación.

Position Mode

El modo de operación de posición, se diferencia del de perfil de posición, en que predomina la posición y no se pueden modificar trayectorias para llegar a esa posición. Este modo de operación se utiliza cuando se quiere llegar a una posición sin importar los demas parámetros, aceleración, velocidad...

Velocity Mode

En el modo de operación de velocidad, se establece como parámetro determinante la velocidad del motor. Este modo es útil para hacer un buen control de velocidad, siempre y cuando se tenga un buen sistema de lazo cerrado para el control de la posición, de esta forma controlaremos la velocidad por un lado, y de forma externa, mediante un bucle, la posición.

Current Mode

En el modo de operación de corriente, funciona ejerciendo un control sobre la corriente, y limitando una velocidad. Este modo se usa cuando se tiene una referencia de posición con un bucle de control de velocidad.

Master Encoder Mode

En el modo de operación de Encoder Maestro, se pretende controlar una posición por un encoder externo. Este modo funciona mediante dos entradas digitales, con las cuales se controla la posición, en este modo se puede cambiar la polaridad del software, por tanto podríamos poner la dirección de giro que se desee. Como se tiene una controladora EPOS2 50/5, las entradas las tenemos en los pines 7 y 8, a continuación se muestra un ejemplo de funcionamiento.

Quadrature Counter: EPOS 70/10, EPOS2 50/5 & EPOS2 Module 36/2		
Channel A	Digital Input 8	
Channel A\	Digital Input 8\	
Channel B	Digital Input 7	
Channel B\	Digital Input 7\	
Digital Position Desired Value (Polarity = 0)		

Figura 11: Master Encoder Mode, Ejemplo de funcionamiento.

Step Direction Mode

En el modo de operación de Dirección de paso, el controlador se comporta como un motor paso a paso, de esta forma se tendrán dos entradas digitales, las cuales una representa la posición y la otra el impulso. En la controladora Epos2 50/5 las entradas se tienen en los pines 7 y 8.

Up/Down Counter: EPOS 70/10, EPOS2 50/5 & EPOS2 Module 36/2		
Step	Digital Input 8	
Step\	Digital Input 8\	
Direction	Digital Input 7	
Direction\	Digital Input 7\	
Digital Position Desired Value (Polarity = 0)		

Figura 12: Step Direction Mode, Ejemplo de funcionamiento.

2.3. Propiedades del objeto

En este apartado, se va a tratar la creación de un objeto, cuyas propiedades sean los parámetros necesarios para el buen funcionamiento de el exoesqueleto. Para la descripción de nuestro objeto, se va a analizar el tipo de datos necesario para el funcionamiento con la librería, se describirá cada una de las variables creadas, y por último se creará una clase con todas estas propiedades y su constructor.

2.3.1. Tipos de datos

En este apartado, se va a definir los tipos de datos que el controlador utiliza. Para un buen funcionamiento se necesita que los tipos de datos que están declarados en la librería, sean los mismos tipos que se le ofrezcan para la configuración. Como se ha comentado en el apartado de Comunicación, se ha tenido que realizar la conversión a su equivalente de un tipo de dato, ya que éste tipo de dato no era compatible con nuestro software, y por tanto se ha hecho su equivalencia de forma satisfactoria.

A continuación se presenta una tabla con los tipos de datos necesarios, y con los rangos de valores que son compatibles, esto hay que tenerlo en cuenta a la hora de darle valor a las variables.

Name	Data Types	Size Bits	Size Bytes	Range	Comment
char, __int8	signed integer	8	1	-128…127	
BYTE	unsigned integer	8	1	0…256	
short	signed integer	16	2	-32'768…32'767	
WORD	unsigned integer	16	2	0…65'535	
long	signed integer	32	4	-2'147'483'648…2'147'483'647	
DWORD	unsigned integer	32	4	0…4'294'967'295	
BOOL	signed integer	32	4	TRUE = 1 FALSE = 0	
HANDLE	pointer to an object	32	4	0…4'294'967'295	Depending on OS
		64	8	0…18'446'744'073'709'551'615	

Figura 13: Tipos de datos, Descripción.

Como se puede ver, en la tabla se especifica cada rango de valores, y también se tendrá en cuenta sus equivalentes sin signo, es decir los tipos de datos unsigned, que se escriben de la siguiente forma(ulong, uint...). Con toda esta información ya se podrá configurar de forma correcta los valores de las variables, para tener una coherencia con los tipos de datos de la librería.

2.3.2. Definición de Propiedades

Para una correcta configuración y comunicación, el objeto creado tiene que tener una serie de propiedades, las cuales tienen que ser compatibles con los tipos de datos de la librería, y crearemos tantas propiedades como parámetros a modificar se deseen.

1. Propiedades de Comunicación. Estas propiedades se utilizan para configurar el tipo y protocolo de comunicación que se va a usar, así como su configuración.

    - Dll → Archivo EposCmd.dll, librería donde están configuradas todas las funciones de la controladora.
    - Header → Archivo Definitions.h, necesario para definir los tipos de datos de la librería.
    - ReturnValue → Propiedad que sirve para comprobar que una instrucción se ha hecho de forma correcta, ya que si se hace de forma incorrecta da un valor diferente.
    - DeviceName → Esta propiedad para este proyecto es 'EPOS2' y es un parámetro de configuración para la comunicación con la controladora.
    - ProtocolStackName → En esta propiedad se establecerá el protocolo utilizado para la conexión, en este caso al ser por USB, el valor de esta propiedad es 'MAXON SERIAL V2'.
    - InterfaceName → Esta propiedad nos indica la interfaz a la que está conectado, en este caso 'USB'.
    - PortName → Esta propiedad sirve para identificar, de forma única el puerto al que está conectado cada controlador, (USB0, USB1,...).
    - KeyHandle → Es un parámetro identificador de tipo HANDLE
    - NodeId → Esta propiedad nos identifica el nodo al que está conectado la controladora.
    - Baudrate → Parámetro de configuración, 1e6
    - Timeout → Parámetro de configuración, 100/500mseName → Esta propiedad nos indica la interfaz a la que está conectado, en este caso 'USB'.

2. Propiedades de Configuración. Estas propiedades son las mínimas necesarias para poder configurar el controlador y poder usarlo.

    - Mode → En esta propiedad se selecciona el modo de operación deseado.

- pMode → Esta propiedad recibe el valor del modo de operación con el cual está configurado.
- ProfileVelocity → Parámetro de configuración del perfil de Velocidad para los modos de operación.
- Connected → Esta es una propiedad, creada para el control de la comunicación, y que se activa cuando todo el proceso de comunicación ha ido satisfactoriamente y se puede proceder al control del exoesqueleto.
- MaxAcceleration → Limitación del parámetro de la aceleración.
- MaxFollowingError → En esta propiedad se establece un error máximo.
- MaxProfileVelocity → Limitación del parámetro de la velocidad.
- VelDimension → Dimensión de las unidades de la velocidad.
- VelNotation → Unidades de la velocidad, rpm.
- P → Parámetro Proporcional del controlador PID
- I → Parámetro Integral del controlador PID
- D → Parámetro Derivativo del controlador PID

3. Propiedades de Motores y Sensores. Estas propiedades nos indican el tipo de motor y sensor que se quiere controlar.

   - MotorType → Parámetro de configuración de Hadware, en el que indica el tipo de motor.
   - NominalCurrent → Parámetro del motor, corriente nominal.
   - MaxOuputCurrent → Parámetro del motor, Máxima salida de corriente.
   - ThermalTimeConstant → Parámetro del motor, Constante Térmica.
   - NbOfPolePairs → Parámetro del motor, número de pares de polos.
   - SensorType → Tipo de sensor que se tiene instalado.
   - EncoderResolution → Resolución del Encoder.
   - InvertedPolarity → En esta propiedad se puede modificar la polaridad del sensor.

2.3.3. Creación de una clase

Se usará Matlab para el control del exoesqueleto, y esto quiere decir que la programación sea orientada a objetos, y por tanto la necesidad de usar una clase es para la definición de propiedades, métodos y eventos.

Para que todas las propiedades o variables descritas antes sean fáciles de asignar, se creará una clase en Matlab. Se creará una clase 'class epos' de tipo handle, la cual tendrá todas las propiedades anteriormente descritas, de esta forma, se podrán crear diferentes objetos para las diferentes controladores con las mismas propiedades, simplemente variando el objeto.

Nuestra utilidad para usar la clase, es crear un constructor que cree un objeto con todas estas propiedades. De esta forma en nuestra rutina para conectar diferentes controladoeas, y como veremos mas adelante, con las mismas líneas de código y funciones, y solo variando unas propiedades se podrá controlar cualquier número de motores y configurar sus controladores.

```
classdef class_epos < handle
    %   MOTOR EC45, MOTOR DE LA CADERA
    properties
        Dll = 'EposCmd'; % 'EposCmd.dll'
        Header = 'Definitions';% 'Definitions.h'
        ReturnValue = 0; % VALOR BOOL QUE NOS DEVUELVEN LAS FUNCIONES DLL
        DeviceName = 'EPOS2';% 'EPOS2'
        VelNotation = 0;
        %RESTO DE PROPIEDADES
        %.
        %.
        %.
        %.
        P = 1000;
        I = 100;
        D = 1000;


    end
    methods
        function epos = class_epos() % constructor
        end
    end
end
```

Figura 14: Ejemplo de la clase.

2.4. Matlab

En esta sección, se va a desarrollar en el entorno de Matlab, una comunicación completa. Para llegar a esa comunicación, esta sección se dividirá en tres: Rutina de conexión, Rutina de operación y Rutina de desconexión.

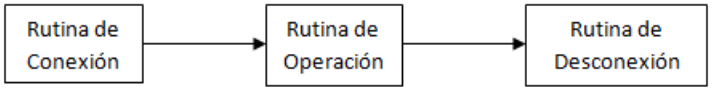


Figura 15: Rutina de funcionamiento en Matlab.

En la Rutina de conexión se tratarán y explicarán los parámetros y funciones de configuración necesarias, hasta llegar a la habilitación para realizar el control del controlador. En la Rutina de operación se va a describir el modo de operación seleccionado, sus funciones para su configuración y se va a proceder a realizar un movimiento. En la Rutina de desconexión, se explicará como en un orden especificado, la desconexión correcta del hadware y las librerías usadas para su funcionamiento.

2.4.1. Rutina de conexión

Para realizar una buena conexión y comunicación con el Hadware, se necesita seguir un orden a la hora de su configuración. En primer lugar se necesita cargar la librería para la comunicación con el Hadware, en segundo lugar se necesita la configuración de motores, sensores y parámetros de operación límite, y por último, si todo ha salido de forma satisfactoria se habilitará el controlador para poder realizar movimientos y control. Por tanto, para una buena rutina se tendrá que seguir estos pasos de forma rigurosa, de no ser así, la configuración y el control puede ser erróneo.

1. Parametrización de la conexión.
2. Configuración de motores, sensores y parámetros límite.
3. Habilitar controlador.

Parametrización de la conexión

En esta parte, se va a desarrollar cada una de las instrucciones necesarias para la comunicación, así como su orden de ejecución para que todo funcione de forma correcta.

1. Loadlibrary. Con la función loadlibrary predefinida por Matlab, lo que se pretende es la posibilidad de cargar un archivo .Dll con su archivo de definiciones .h . Esta función se usará al inicio de la rutina siempre, ya que se necesita en todo momento, puesto que es la librería que contiene todas las funciones y definiciones necesarias para el controlador.

2. OpenDevice. En esta parte, ya se pasa a las funciones de la librería para la conexión con el controlador. En la función OpenDevice, lo que se hace es abrir el puerto para una primera configuración del controlador. Para que esta función funcione de forma correcta se necesita que sus parámetros de entrada estén definidos de forma correcta. Estos parámetros son:

   - DeviceName
   - ProtocolStackName
   - InterfaceName
   - PortName

3. SetProtocolStackSettings. En esta función se establecerá los parámetros de comunicación. Estos parámetros son la velocidad de transferencia y el tiempo que está abierto el puerto, para eso, esas propiedades del objeto tienen que estar definidas anteriormente. Los parámetros son:

   - Baudrate
   - Timeout

4. ClearFault. Esta función, aunque no necesaria, es muy importante ponerla en esta posición de la rutina de comunicación, debido a que si hubiera algún error guardado en la controladora, esta función lo quitaría y nos dejaría trabajar, de lo contrario, el control queda bloqueado. Los parámetros de entrada de esta función son:

   - NodeId

5. SetPositionRegulatorGain. Esta función establece los parámetros PID del controlador de posición integrado en el controlador. Estos parámetros han sido establecidos por nosotros previamente mediante un ensayo experimental. Los parámetros de entrada son:

- NodeId
- P
- I
- D

6. SetVelocityUnits. Con esta función estableceremos las unidades de los parámetros de velocidad, así como su notación. En la notación podemos elegir si es estándar, deci-, centi-, o mili-.
   - NodeId
   - VelDimension
   - VelNotation

Configuración de motores, sensores y parámetros límite

En esta parte, se establecerá el tipo de motor y sensor que estamos utilizando, así como sus características. También se establecerá los parámetros límite de operación. Las características del motor y del sensor, se encontrarán en el catálogo del fabricante. En esta parte, es donde se van a modificar los parámetros en el caso de que se modifiquen los motores o sensores, se tienen dos clases preparadas con las configuraciones para los motores EC45 y EC60. Para realizar todo esto se procederá con las siguientes instrucciones:

1. SetMotorType. Esta función permite establecer de que tipo es el motor el cual se está usando, ya que el fabricante dispone de tres tipos de motores. En este caso se usará el tipo 10, que corresponde con un motor EC senoidal conmutado. Los parámetros para esta función son:
   - NodeId
   - MotorType
2. SetEcMotorParameter. En esta función se van a establecer los parámetros de operación del motor. Con estos parámetros se limitará la máxima corriente en continua, la máxima corriente admitida de salida, la constante térmica y el número de pares de polos que tiene el motor.
   - NodeId

- NominalCurrent
- MaxOutputCurrent
- ThermalTimeConstant
- NbOfPolePairs

3. SetSensorType. Como se ha hecho anteriormente con el motor, ahora con esta función se establecerá el tipo de sensor que tenemos montado. Se puede elegir entre 5 tipos de sensores diferentes, pero en este caso, se está utilizando un sensor incremental de dos canales. Los parámetros de entrada de esta función son:

   - NodeId
   - SensorType

4. SetIncEncoderParameter. Como se trata de un encoder incremental, para establecer su modo de funcionamiento se usará esta función. Con esta función se podrá establecer la resolución del encoder, y se podrá elegir la polaridad con la que trabajará el sensor. La máxima resolución del encoder se encuentra en el catálogo del producto. Los parámetros de configuración son:

   - NodeId
   - EncoderResolution
   - InvertedPolarity

5. SetMaxAcceleration. Con esta función, se va a establecer un criterio de seguridad con respecto a la máxima aceleración y deceleración permitida. Esta función limita la aceleración a la hora de generar trayectorias en los modos de operación. Los parámetros de la función son:

   - NodeId
   - MaxAcceleration

6. SetMaxFollowingError. Esta función, es también una función de seguridad, debido a que limita el error máximo que se puede tener, entre la posición demandada y la actual. Esta propiedad se mide en pasos, en este caso, el error definido es de 2000 pasos. Los parámetros de entrada son:

   - NodeId

- MaxFollowingError

7. SetMaxProfileVelocity. Con esta función, se establecerá un límite de seguridad para el perfil de velocidad, de esta forma no podrá superar esta velocidad el motor. Este límite afecta a las trayectorias de los diferentes modos de operación. Los parámetros de entrada de la función son:

   - NodeId
   - MaxProfileVelocity

8. SetOperationMode. Antes de poder habilitar la controladora, se tiene que establecer el modo de operación con el que va a trabajar. Como se ha visto en su capítulo tenemos diferentes modos, aunque en este caso, se usará el modo de perfil de posición. Los parámetros de entrada de la función son:

   - NodeId
   - Mode

Habilitar controlador

Por último, para acabar con la rutina de comunicación, solo queda habilitar la controladora para poder acceder al control del exoesqueleto. La controladora no se va a poder habilitar si no se han hecho correctamente los pasos anteriores. Si toda lo configuración ha sido correcta se procederá a habilitar la controladora con la siguiente instrucción:

1. SetEnableState. Con esta función se va a habilitar la controladora que corresponda con el objeto tratado. Los parámetros para la ejecución de la función son:

   - NodeId

2.4.2. Rutina de operación

En esta fase, se va a desarrollar las funciones que se han utilizado para activar un modo de operación, para mover las articulaciones y para la recepción de datos del sensor. En los modos de operación se podrán modificar sus parámetros cada vez que se desee realizar un movimiento. En este caso, se podrá modificar la trayectoria del perfil de posición en cada movimiento, modificando así su velocidad, aceleración... En la rutina de operación, se podrán activar diferentes modos de operación, pero los parámetros de cada modo de operación, solo se podrán cambiar siempre que el modo esté activado.

Establecer modo de operación

En esta parte se va a desarrollar la función utilizada para la activación del modo de operación. En este caso, al querer trabajar con el modo de operación de perfil de posición, se tendrá una función específica para activar este modo. Para cada modo de operación se dispone de una función específica de cada uno par la activación de éste.

1. ActivateProfilePositionMode. Esta función únicamente permite activar el modo de operación, de perfil de posición, no pudiéndola usar para activar cualquier otro modo. Los parámetros necesarios para su funcionamiento son:

    - NodeId

Movimiento de articulaciones

En esta parte se va a desarrollar como se configuran los parámetros de un modo de operación y el movimiento de los motores. Como se ha mencionado antes, existen mas modos de operación pero en este caso se va a trabajar con el modo de perfil de posición, por lo tanto, las funciones aquí descritas únicamente serán válidas para este modo de operación. En este modo de operación, se podrán modificar los perfiles de aceleración, de velocidad, y por consiguiente la trayectoria. También se podrá modificar si se quiere una trayectoria trapezoidal o senoidal. Con respecto al movimiento, se establecerá si se quiere mover el motor en coordenads relativas o absolutas, la posición a la que se desea mover el motor y si se quiere que el movimiento del motor sea instantáneo, o espera a que se realice el movimiento anterior para realizar el nuevo. Todo esto se configura con las siguientes funciones:

1. SetPositionProfile. En esta función se van a establecer los datos de configuración del modo de operación. Se podrá configurar la velocidad, la aceleración, y por consiguiente la trayectoria generada por ambos. Los parámetros de entrada de la función son:
   - NodeId
   - ProfileVelocity
   - ProfileAcceleration
   - ProfileDeceleration
2. SetObject. Esta función sirve para modificar cualquier objeto de la controladora, en este caso la usaremos para modificar el perfil de trayectoria, para que sea senoidal. Los parámetros de entrada de la función son:
   - NodeId
   - ObjectIndex
   - ObjectSubIndex
   - pData
   - NbOfBytesToWrite
3. MoveToPosition. Esta función es la encargada del movimiento del motor, y en ella se podrá definir la posición deseada del motor, el sistema de coordenadas, si es absoluto o relativo, y si el movimiento se quiere que se realice de forma instantánea.
   - NodeId
   - TargetPosition
   - Absolute
   - Inmediately

Lectura del sensor de posición

En la parte de recepción de datos del sensor, se realiza de forma simple mediante una función. La lectura se recibe en pasos, por lo tanto si se está trabajando en otra unidad, se tendrá que convertir para que sea coherente con el proyecto. La función usada para la lectura del encoder es:

1. GetPositionIs. Esta función es la encargada de la lectura de los datos del encoder, estos datos vienen predefinidos por los parámetros de configuración del encoder. Los parámetros de entrada de la función son:
    - NodeId
    - PositionIs

2.4.3. Rutina de desconexión

Por última, y para finalizar la conexión y comunicación de forma exitosa, se necesita una buena rutina de desconexión, la cual se tiene que hacer de forma ordenada y secuencialmente, de esta forma se asegurará una buena desconexión. En primer lugar se deshabilitará la controladora, des pues se preocederá al cierre de la comunicación con el ordenador y por último se quitará la librería de la memoria del ordenador.

Deshabilitar controladora

En primer lugar se deshabilitará la controladora, si no fuera de esta forma, si antes se cerrara la conexión o quitáramos la librería no se podría deshabilitar, creando un conflicto en la siguiente conexión. La función que realiza esta tarea es:

1. SetDisableState.

Desconexión de dispositivos

Una vez deshabilitada la controladora, se procederá al cierre de la comunicación, y esto se va a hacer mediante dos funciones, una que cierra los dispositivos uno a uno y otra que los cierra todos, de esta forma se asegura una desconexión correctta. Las funciones son:

1. CloseDevice.
2. CloseAllDevices.

Quitar libreria

Por último, para finalizar la conexión se quitara de memoria la librería mediante una función de Matlab. Esta función es unloadlibrary, y como parámetro sería la librería que se está usando.

2.5. Simulink

Para la simulación y pruebas, se va a usar el software de simulación de Matlab, Simulink. Simulink es un módulo de Matlab el cual permite una comunicación en tiempo real entre la computadora y la controladora. El principal objetivo es realizar una comunicación en tiempo real, en la cual se pueda controlar el exoesqueleto, y la recepción de los valores leídos por los sensores de cada uno de los motores. Para la implementación, se va a usar un diagrama de bloques muy simple.

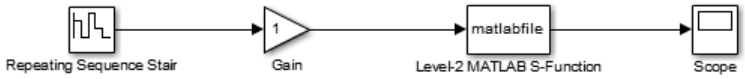


Figura 16: Diagrama de bloques genérico.

Como se observa en la figura, principalmente se va a trabajar con cuatro tipo de bloques diferentes. El bloque que realmente se va a encargar de toda la tarea de comunicación y control es el bloque de Level-2 Matlab S-Function. Se va a utilizar una entrada en escalón y un tren de pulsos para las pruebas realizadas, y para ver la respuesta se utilizará un osciloscopio. A continuación se va a detallar la función de cada bloque:

- Source. Este bloque sirve para darle una entrada al sistema de control, es decir, para decirle a que posición se desea que vaya el motor. El bloque de Source en este caso, principalmente será un tren de pulsos, que se modificará de tal forma, para ver la reacción de los motores ante entradas muy dispares, y para de esta forma, poder sintonizar los valores del controlador de una forma coherente.

- Gain. Este bloque representa una ganancia, principalmente se pone este bloque para controlar mediante este parámetro una constante proporcional. En este caso este bloque se usa para la dirección de los motores, es decir, para que todos los motores tengan el mismo sistema de giro, simplificando al máximo la rutina de control y operación.

- Level-2 Matlab S-Function. Este bloque, será el más importante de todos, ya que permitirá la inclusión de script de Matlab en el diagrama de bloques. Este bloque crea una instancia al modelo, en la cual se puede introducir parámetros. Para su utilización, el blo-

que tiene que estar configurado con el nombre del script, y con los parámetros necesarios de configuración.

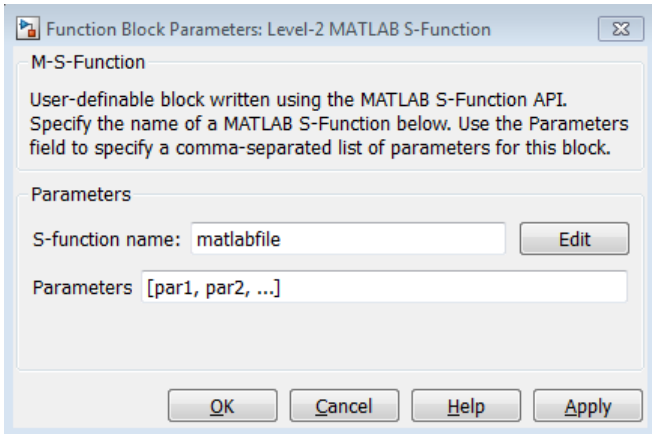


Figura 17: Condiguración Level-2 Matlab S-Function.

- Scope. En este bloque se va a representar en un osciloscopio digital, la posición deseada y la posición en cada instante de tiempo de los motores, leyendo la información del encoder que llevan éstos instalados.

2.5.1. Diagrama de bloques

El diagrama de bloques para este proyecto, será muy similar al planteado en el apartado anterior, añadiendo un bloque más, que será el que gestione la conexión con la librería. Se va a desarrollar el modelo para 1 motor, el cual sería el siguiente:

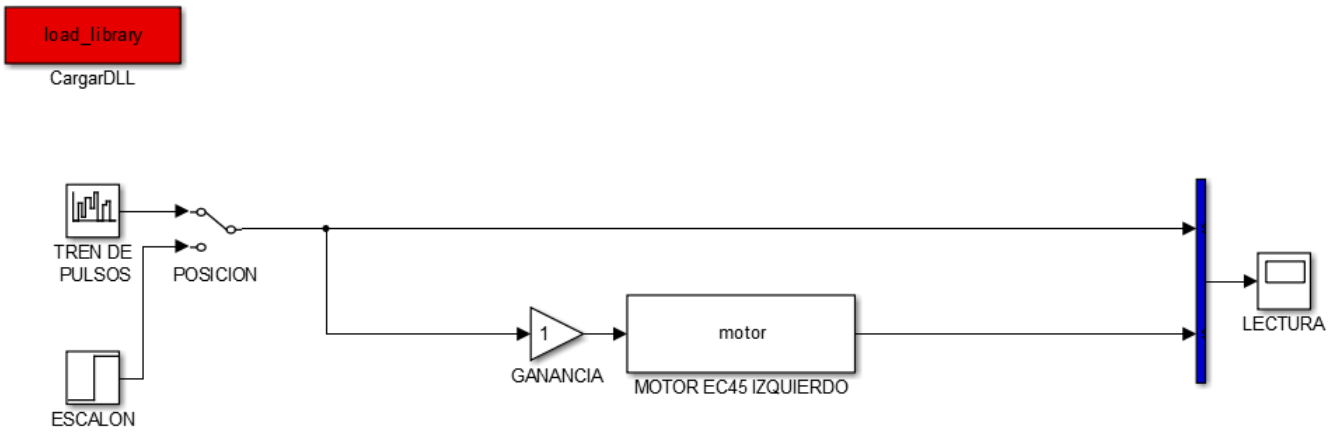


Figura 18: Diagrama de bloques genérico.

Como se puede apreciar en el diagrama de bloques, se tienen dos partes diferenciadas, pero que a su vez dependen entre sí. Se tendrá una primera parte que es un bloque aislado del resto,

en el cual se encargará de cargar la librería, y en segundo lugar, se tendrá un diagrama de bloques más completo que se encargará de toda la rutina de comunicación con la controladora, movimiento, lectura de datos y de desconexión.

Cargar Libreria

En primer lugar, lo que se tiene que asegurar es que se haya cargado la librería antes de que la rutina de comunicación funcione, y por eso se tratará de forma independiente de la rutina de comunicación. Para hacer la carga de la librería y que ésta condicione al resto, se va ha crear una variable global, que se va a encargar de que si la librería está cargada funcione el modelo, y si no, que emita un error, o que hasta que no se cargue no funcione.

Esta función, principalmente lo que hace, es cargar la librería con la función de Matlab 'loadlibrary', y cuando se ejecute, se comprobará mediante la función 'libisloaded' que la librería está correctamente cargada. Una vez comprobada la carga de la librería, la variable global gLibraryLoad, se establecerá a 1(si ha fallado), o a 2(Si está cargada correctamente). Por último para hacerlo de una forma más visual, mientras la librería esté sin cargar, el fondo del bloque estará de color rojo, y en el momento que se cargue se pondrá de color verde, de esta forma, se podrá saber de forma visual si la librería está cargada o no.

(a) Librería sin cargar

(b) Cargando Librería

(c) Librería cargada

Rutina de Operación

En la rutina de operación se va a realizar todas las tareas de comunicación, configuración, movimiento, control, y desconexión de cada uno de los motores. Se han agrupado todas estas tareas en una rutina, debido a que los motores pueden variar, las controladoras también, y de esta forma se tendrá de forma independiente cada controladora, junto con sus motores y sensores. Esto es muy importante, ya que en cada momento se va a tener un control total de cada motor por separado, pudiendo variar sus características de operación en cada momento, sin que afecten a los otros motores. La rutina de operación se va a dividir en 3 partes: comunicación,

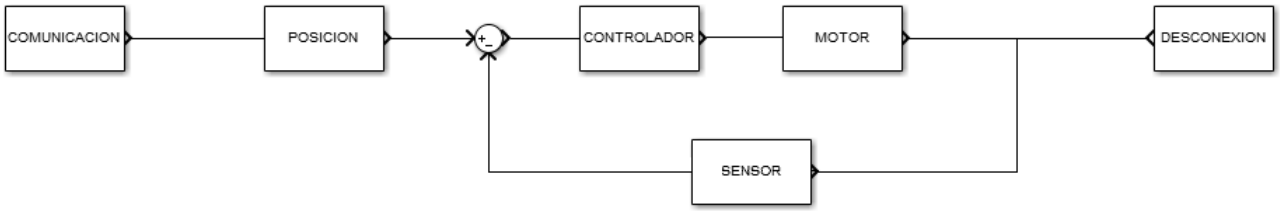


Figura 19: Rutina de operación completa.

control y desconexión. Esta rutina está controlada principalmente por el bloque y el script 'motor.m'. Este bloque tiene unos parámetros de configuración, los cuales son el puerto al que está conectado la controladora, y el tipo de motor. Con estos parámetros se permite crear un objeto, el cual tiene las propiedades de ese motor y la comunicación se va ha hacer en ese puerto. Para habilitar las tareas de comunicación y configuración de la controladora, como se ha mencionado anteriormente, se dispone de una variable global de control gLoadLibrary, la cual se encarga de asegurar que la librería esté cargada. Si la librería está cargada, se dispone a realizar el protocolo de comunicación y configuración de la controladora. Este protocolo se divide en varias partes: comunicación, configuración de motores, sensores y parámetros límite, y por último la habilitación de la controladora.

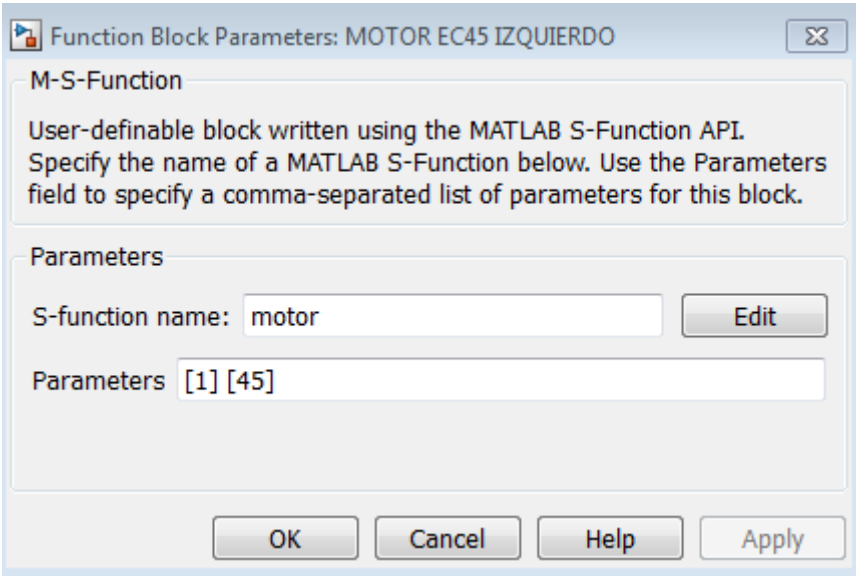


Figura 20: Parámetros del bloque S-Function.

Toda la rutina de comunicación se encuentra en un archivo llamado 'library-epos.m'. En este archivo se encontrará con una serie de opciones relacionadas con la comunicación y configuración

de la controladora. En primer lugar realizaremos el protocolo de conexión y comunicación, en el cual se abrirá el puerto y se establecerá un protocolo. Después se procederá a la configuración de todos los parámetros referentes al motor, sensor y parámetros límites de seguridad que se deseen. Una vez que toda la parte de configuración ha sido realizada de forma correcta, se procederá a habilitar la controladora para poder realizar el control del exoesqueleto. Cuando se habilita la controladora, automáticamente, y si no se ha producido ningún error, una variable global lo recoge, produciendo así una segunda variable de control global, la cual se encarga de que la rutina de control se pueda hacer sin ningún tipo de problema.

Para el movimiento del motor, necesitamos que nos lo habilite la variable gEposConnected. Esta variable se tratará como una propiedad del objeto, teniendo cada objeto creado su propia variable de control. Una vez que la variable nos habilita el movimiento del motor, se procederá a leer la posición en la que se encuentra, y si es diferente de la que se desea, se moverá el motor, teniendo esta rutina en un script llamado 'movements.m'. En esta rutina de movimiento, se establecerá una conversión de grados a pasos del motor, se establecerá la velocidad y aceleración, con la que se creará una trayectoria. Se configurará para tener una trayectoria o de tipo trapezoidal o de tipo senoidal, y una vez configurado se procederá al movimiento, configurando éste en coordenadas absolutas o relativas, y si queremos el movimiento instantáneo o esperamos a que acabe el movimiento anterior.Después de realizar el movimiento, se leerá la posición en la que se encuentra el motor en cada instante de tiempo, pudiendo mediante esta señal, actuar y hacer acciones de control del tipo PID.

Después de realizar toda la rutina de operación necesaria, se procede con la rutina de desconexión, la cual, en un orden secuencial, deshabilita la controladora, cierra conexión con los diferentes dispositivos conectados al puerto, y por último quita la librería de memoria. Después de toda esta rutina, el exoesqueleto está preparado para una nueva rutina de operación.

3. Resultados

3.1. Conexión y Comunicación

Como primera parte del proyecto, se pidió una conexión y comunicación entre las controladoras y el software Matlab. Para ello se ha implementado un protocolo de comunicación antes descrito, en el cual se establece una conexión mediante USB entre las controladoras y Matlab o Simulink.

Esta primera parte consiste en realizar una conexión entre Matlab con las controladoras, ya que no se dispone de drivers para este software. Para realizar la conexión se ha partido de un archivo .DLL modificado para C++ y de un archivo Definitions.h, estos archivos fueron proporcionados por una persona que realizo la conexión con Matlab y estas controladoras. Los archivos fueron probados y no funcionaron ya que no estaban correctamente configurados y los tipos de datos no eran válidos para Matlab. Se modificó el archivo Definitions.h haciendo compatible la librería para Matlab. Una vez conseguida una librería válida para la conexión con Matlab se procede a establecer un protocolo de conexión y comunicación.

Cada controladora se conecta al ordenador mediante un USB diferente, de esta forma el identificador del puerto hay que ponerlo como parámetro en cada controladora. Para que se pueda configurar de una manera eficiente cualquier configuración de motores, se dispondrá de una o varias clases con los diferentes motores y actuadores que se pudieran poner en nuestra controladora, de esa forma solo se configurará la parte de la controladora y de comunicación, teniendo cada uno de los parámetros de motores y sensores en sus clases correspondientes.

En todo el protocolo de comunicación se configurará cualquier parámetro que tenga que ver con la conexión, así como todos los parámetros de seguridad. Se establecerán los límites máximos de aceleración, de velocidad y de error permitido. Después de configurar parámetros de seguridad y de comunicación, se procederá a la configuración de los parámetros de los motores y sensores que se vayan a controlar, para eso, y como se ha mencionado anteriormente, se dispondrá de una o varias clases con estos datos predefinidos. En última instancia se establecerá el modo de operación deseado, y por último se procederá a la habilitación de la controladora para proceder al control del motor y movimientos. Para que la controladora se habilite tiene que estar correctamente configurada, de lo contrario se pueden producir errores que impedirán que se habilite. Para ver en todo momento que pasa en la comunicación se han establecido unos mensajes por pantalla en los que se informará del proceso que está ocurriendo.

```
Conectando...
Configurando motores y sensores...
Habilitando...
EPOS2 Habilitada
```

Para terminar con la comunicación, es necesario realizar una desconexión de forma correcta para evitar posibles fallos en una comunicación futura. Para realizar una desconexión exitosa, en primer lugar hay que deshabilitar la controladora, en segundo lugar se cerrará la conexión con los puertos USB usados, y por último quitaremos la librería de memoria de nuestro ordenador. Por este orden y de esta forma se procede a una correcta desconexión de la controladora con Matlab.

3.2. Sintonización de PID

En este apartado, se va a sintonizar de manera experimental el PID correspondiente al modo de operación de perfil de posición. Para su sintonía se parte de un modelo de funcionamiento de un motor, y se necesitará que nuestro sistema responda de manera estable y segura, a las diferentes entradas que se puedan tener. Para esto se va a realizar la sintonía mediante una entrada de 10 posiciones diferentes que varían entra 0 y 27 grados.

3.2.1. Control mediante PID

Un controlador PID es un mecanismo de control por realimentación ampliamente usado en sistemas de control industrial. Este calcula la desviación o error entre un valor medido y un valor deseado.

El algoritmo del control PID consiste de tres parámetros distintos: el proporcional, el integral, y el derivativo. El valor Proporcional depende del error actual. El Integral depende de los errores pasados y el Derivativo es una predicción de los errores futuros. La suma de estas tres acciones es usada para ajustar al proceso.

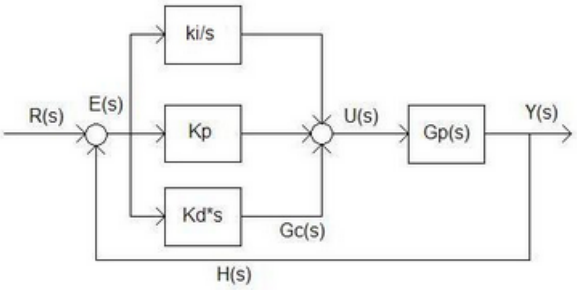


Figura 21: Esquema de funcionamiento PID

Para el correcto funcionamiento de un controlador PID que regule un proceso o sistema se necesita, al menos:

- Un sensor, Encoder situado en el motor.
- Un controlador, EPOS2 50/5.
- Un actuador, Motor.

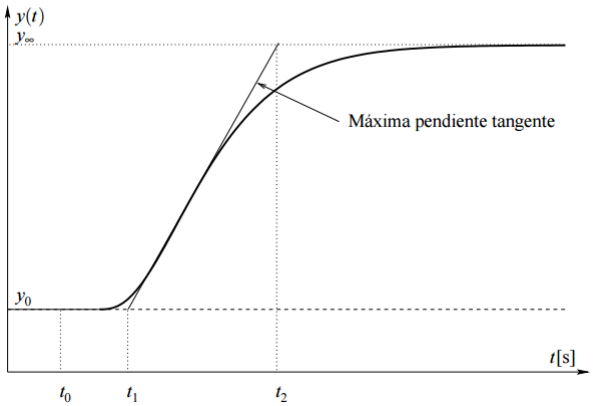
En este caso, se tendrá como esquema de control, un bucle situado dentro de la controladora, la cual lleva implementado un PID que se encarga del control. Para configurar este PID se hace antes de su funcionamiento. El funcionamiento del PID dentro de la controladora solo se puede aplicar en el modo de operación de perfil de posición, ya que para otros modos de operación lleva implementado otro tipo de controlador. En el interior del controlador, se interpreta la señal del encoder, y se contrasta con la señal final, de esta forma se tiene la señal de error, que es procesada mediante el PID, y éste se encarga de enviar las condiciones de trabajo hacia el motor. Para la sintonización del PID, se ha hecho de forma teórica y luego se ha optimizado de forma empírica.

3.2.2. Sintonizado teórico

Para la sintonización teórica, se ha optado por el Método de la curva de reacción de Ziegler-Nichols. Este método consiste en realizar una prueba en lazo abierto con las siguientes características.

1. Llevar manualmente la planta a lazo abierto a un punto de operación normal manipulando u(t). Supongamos que la planta se estabiliza en y(t) = y_0 para u(t) = u_0
2. En un instante inicial t_0 aplicar un cambio escalón en la entrada,u_0 a u_∞ del valor nominal.
3. Registrar la respuesta de la salida hasta que se estabilice en el nuevo punto de operación.
4. Trazar una recta en el punto de máxima pendiente tangente a la curva y sacar los parámetros característicos cuando ésta corte a y_0 e y_∞
5. Obtención de los siguientes parámetros.

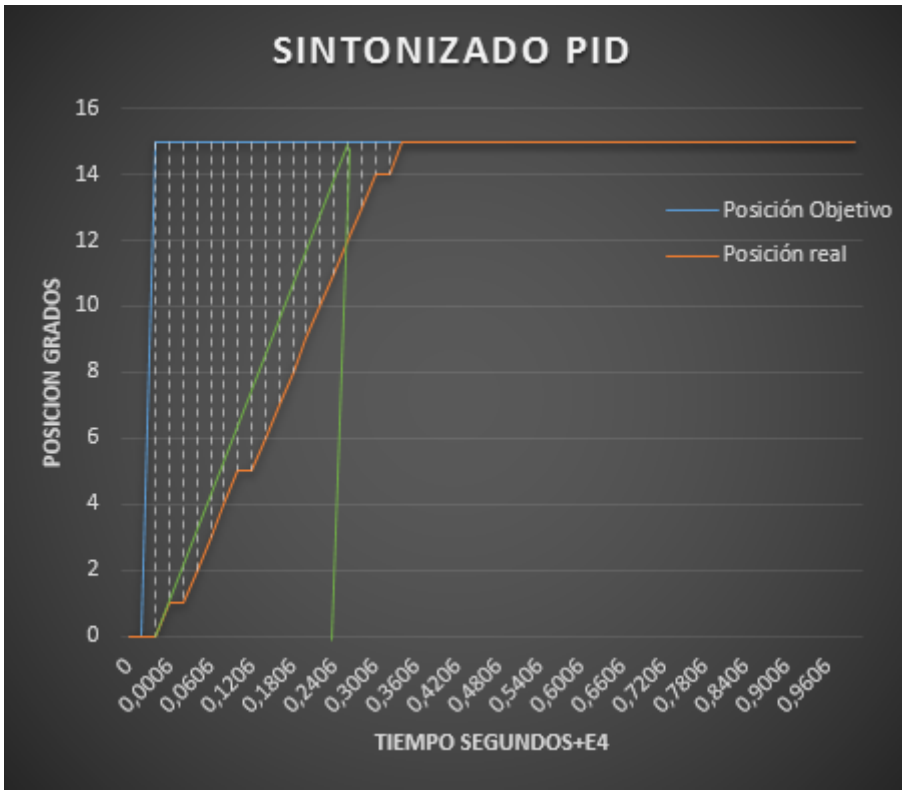
$$k_0 = \frac{y_\infty - y_0}{u_\infty - u_0}, \quad \tau_0 = t_1 - t_0, \quad \gamma_0 = t_2 - t_1.$$



6. Cálculo de los siguientes parámetros.

	K_p	T_r	T_d
P	$\frac{\gamma_0}{K_0\tau_0}$		
PI	$\frac{0{,}9\gamma_0}{K_0\tau_0}$	$3\tau_0$	
PID	$\frac{1{,}2\gamma_0}{K_0\tau_0}$	$2\tau_0$	$0{,}5\tau_0$

Para la sintonización teórica se ha realizado el ensayo anteriormente descrito para un valor objetivo de 15^{o} y con una carga de 5Kg. Los resultados son los siguientes:



Los parámetros obtenidos para aplicar el método de Ziegler-Nichols son:

- κ_0=1; γ_0=2600; τ_0=6

Una vez realizados todos los cálculos necesarios, los parámetros teóricos del PID son:

Proporcional=520 ; Integrador=12 ; Derivativo=3

3.2.3. Sintonizado experimental

Para el sintonizado experimental del PID se partirá de los parámetros calculados anteriormente. El criterio a seguir es la suavidad en la respuesta y el tiempo mínimo de establecimiento. Para conseguir los valores óptimos se va a seguir una pruba de un tren de pulsos en el cual se le aplicará una carga, y se verá la reacción del controlador.

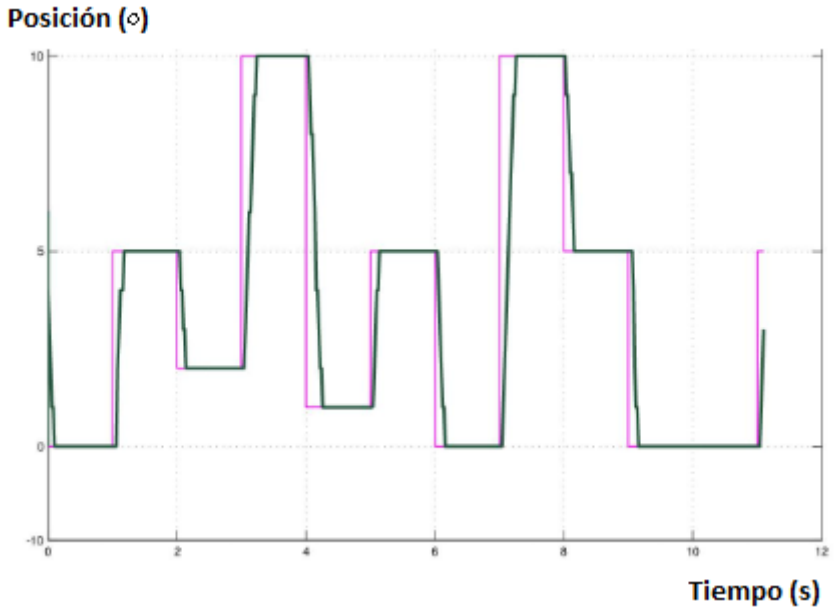


Figura 22: Respuesta con controlador P=520, I=12 y D=3.

En la figura 22 se observa que se tiene una respuesta correcta ante una entrada de pulsos, pero también se observa que no es una respuesta muy suavizada, por lo tanto, se optimizarán los parámetros del PID para conseguir que la respuesta a esta entrada sea lo más suavizada posible y lo más exacta posible. A continuación se realizaran unas pruebas para conseguir los parámetros óptimos.

En la figura 23 se observa una respuesta ante la misma entrada de pulsos y la misma ganancia más suavizada, esto se debe a que se han modificado los valores del parámetro integrador y del parámetro derivativo. También se observa que se necesita una mejor respuesta y para ello se va a modificar el parámetro proporcional.

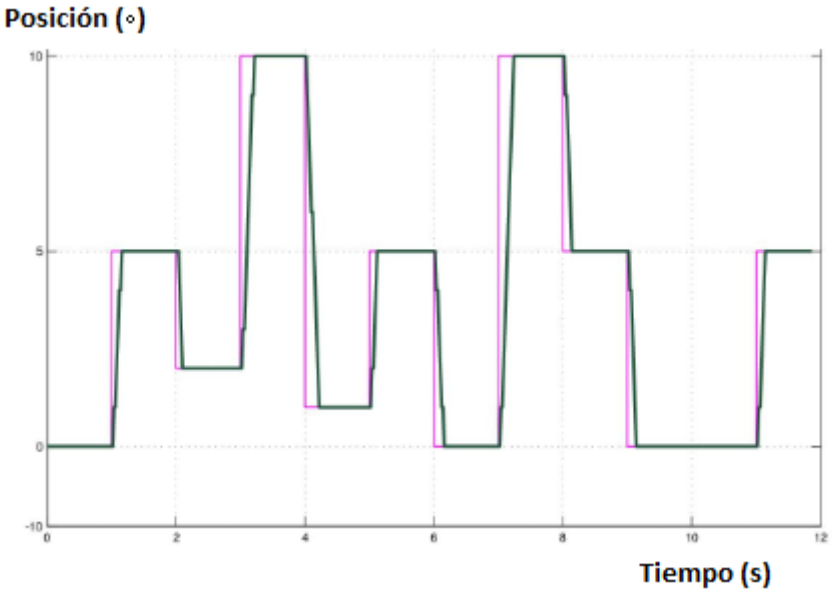


Figura 23: Respuesta con controlador P=520, I=100 y D=100.

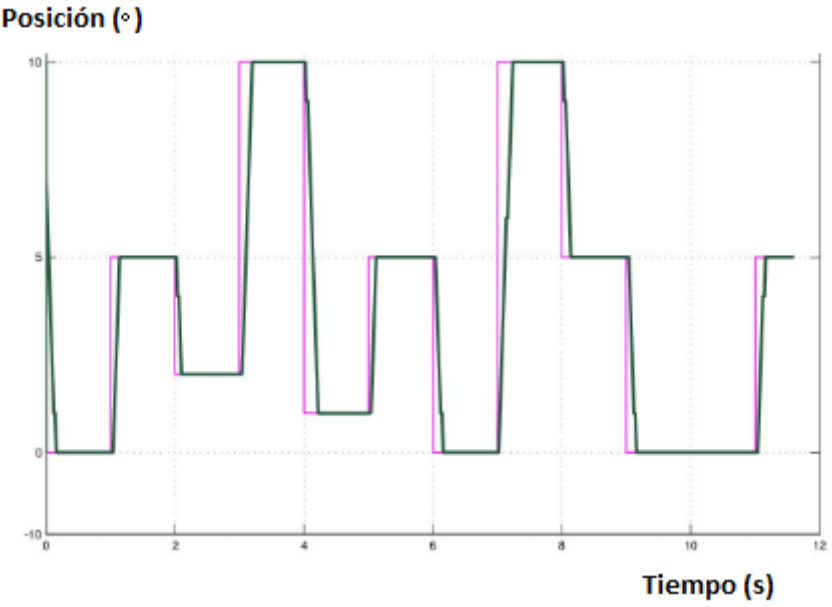


Figura 24: Respuesta con controlador P=1000, I=100 y D=100.

En la figura 24 se observa que se ha suavizado la respuesta del motor ante la entrada de pulsos de diferentes amplitudes. Se considerarán estos parámetros para la sintonización del controlador PID interno. Por tanto, los parámetros del controlador PID son:

Proporcional=1000 ; Integrador=100 ; Derivativo=100

3.3. Movimiento de un motor

Después de haber realizado un protocolo de comunicación exitoso, y disponer de una librería compatible con Matlab, se dispondrá a la segunda parte del proyecto, la cual se corresponde con el movimiento de los motores del exoesqueleto, es decir con el control y movimiento de las articulaciones del exoesqueleto.

Para realizar el movimiento del motor, se ha hecho mediante código en Matlab y Simulink. La diferencia entre ambos, pese a ser el mismo lenguaje de programación, es que Simulink no acepta todas las palabras reservadas e instrucciones de Matlab, y esto hay que adaptarlo a el diagrama de bloques de Simulink. Básicamente, para poder mover y controlar un motor, el procedimiento seguido es el siguiente:

1. Activar modo de Operación.
2. Configurar parámetros del modo de Operación.
3. Realizar acción sobre el motor.

Como se ha mencionado anteriormente en el apartado de Modos de Operación, se dispone de múltiples modos de Operación para la controladora. En el caso de este proyecto y para realizar las pruebas se ha usado el modo de operación de perfil de posición. En primer lugar se tiene que activar dicho modo de operación. Una vez activado se procede a establecer los parámetros de configuración del modo, configurables en cada movimiento, estos parámetros son:

- Velocidad
- Aceleración
- Tipo de perfil de la trayectoria
- Posición relativa o absoluta

Una opción muy interesante, es la de poder cambiar el perfil de trayectoria creada, y se puede elegir entre seonidal o trapezoidal. Para realizar movimientos de una forma más realista, se elige una trayectoria senoidal,ya que el perfil de la velocidad es máximo en el recorrido, y mínimo en los extremos. Una vez configurado todo, se realizará el movimiento del motor, y para ello vamos a usar el siguiente modelo en Simulink:

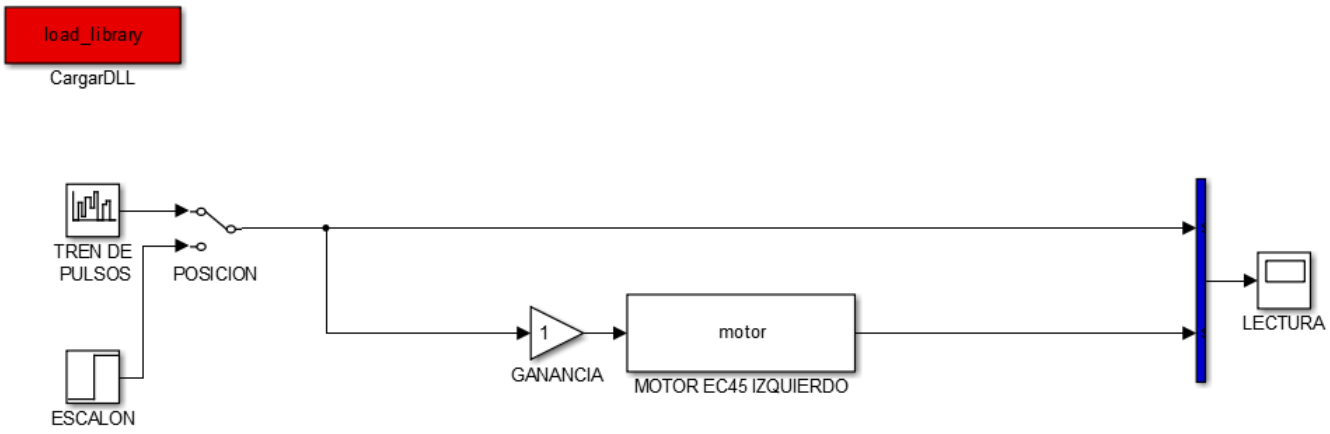


Figura 25: Modelo de 1 motor en Simulink

Como se puede observar en el modelo de Simulink, se dispone de varios bloques en los que se realizarán todas las tareas anteriormente descritas. Para visualizar los datos se usará el bloque LECTURA, que mostrará la posición demandada y la leída por el encoder en cada momento. Para realizar las pruebas, se va a someter al motor a un movimiento de 0-27 grados, en los que se va a generar un tren de pulsos y se verá el comportamiento del mismo.

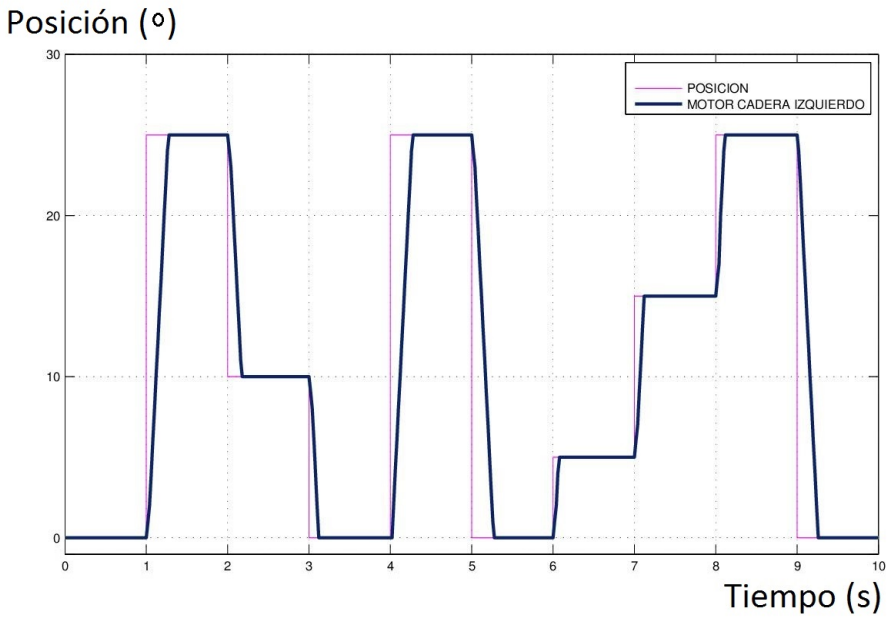


Figura 26: Movimiento de un Motor ante un tren de pulsos.

En la figura 26 se lleva a cabo el movimiento de un motor. En esta gráfica se puede observar y analizar la respuesta del motor ante una entrada de un tren de pulsos. El motor analizado se trata de un EC45, correspondiente a la articulación de la cadera. Como se observa en la figura 26, el motor responde de una manera lineal y eficaz ante las diferentes variaciones de posición dadas. El tiempo que tarda el motor en llegar a la posición dada se debe a la baja velocidad con la que se están realizando las pruebas ya que en su aplicación real la velocidad tiene que ser lo mas parecida a la del movimiento del cuerpo humano. Se puede concluir que con la configuración dada para la conexión y control de un motor, éste funciona de manera aceptable.

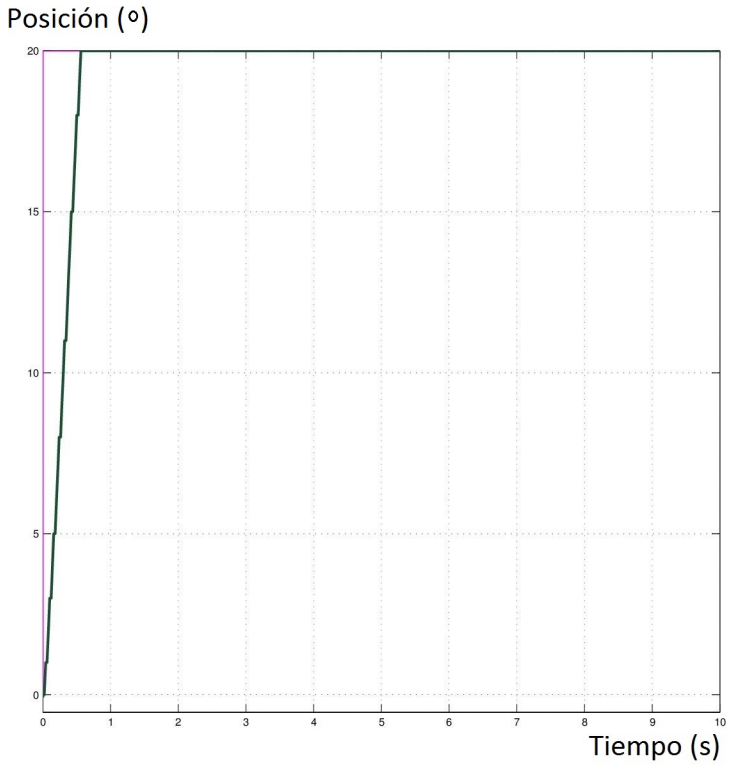


Figura 27: Movimiento de un Motor ante un escalón.

En la figura 27 se observa la respuesta de un motor EC45 a una entrada de tipo escalón. Como se puede observar en la figura, se propone un escalón de 20^{o} el cual es alcanzado por el motor en un tiempo de 0.6 segundos. La respuesta observada es de tipo lineal y de una forma eficaz y segura para la velocidad dada.

3.4. Movimiento simultáneo de dos motores

Para realizar el control sobre dos motores de forma simultánea, el principal problema que se encontró fue la sincronía entre ambos para poder realizar todas las tareas de forma simultánea. Para la realización de esta prueba se ha usado el siguiente modelo en Simulink:

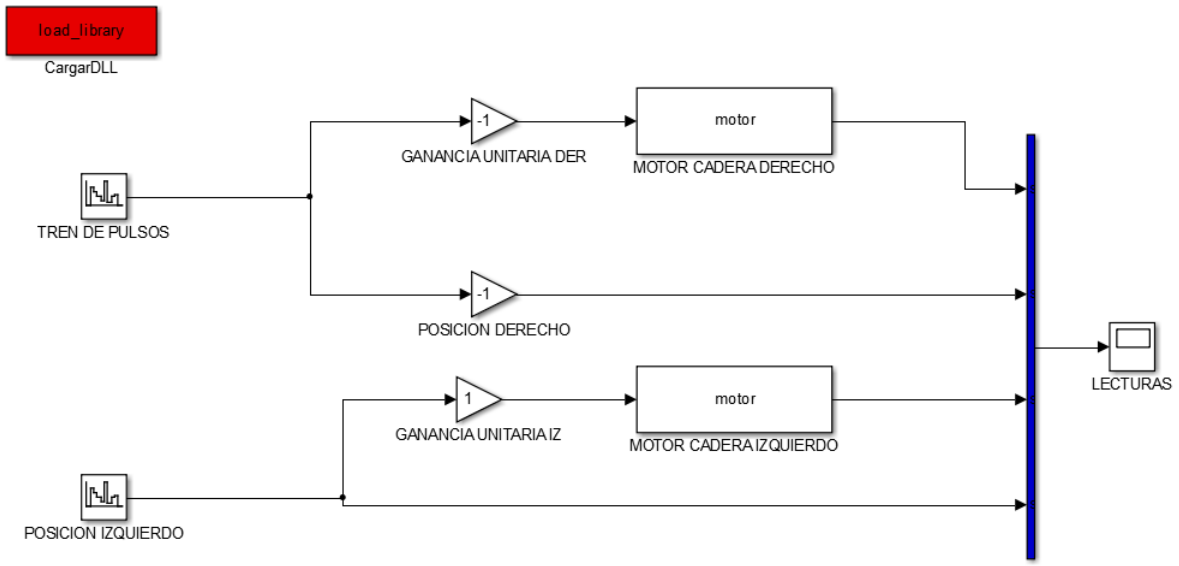


Figura 28: Diagrama de bloques de dos motores simultáneos.

Como se puede observar en el diagrama de bloques, se tiene un motor con coordenadas positivas, y otro con coordenadas negativas. Esto se hace debido a que los motores son los de la cadera, y como están de forma simétrica pero en diferentes sentidos, las coordenadas serán las mismas pero complementarias entre sí. Se va a realizar el estudio con un tren de pulsos que oscilará entre 0 y 27 grados. Como se ha mencionado anteriormente, el principal problema ha sido realizar un código totalmente complementario entre sí, siendo capaz de agregar hadware nuevo o quitarlo sin necesidad de modificar el código. Para ello se ha usado un bloque S-Function Level 2, el cual se comporta como un bucle, inicializa, realiza la operación hasta que se acabe cualquier solicitación, y por último termina.

El bloque principal es el llamado "motor", y en él se ejecuta toda la rutina de configuración de la controladora y la conexión. En este bloque se establece una condición para que empiece a funcionar, y es que la librería tiene que estar cargada, si no está cargada no funcionaría nada. Como se ha dicho, lo importante para este proyecto, es realizar un código fácilmente configurable y modificable a la hora de añadir o quitar hadware a nuestro modelo, y para ello se dispone de una estructura de código en la cual solo se necesitan dos parámetros de

entrada para la configuración de ese hadware. Estos parámetros son: el identificador del puerto al que está conectado y el tipo de motor que tenemos. Con estos parámetros correctamente configurados, el código genera un objeto único de cada bloque, que a su vez va relacionado con el identificador del objeto.

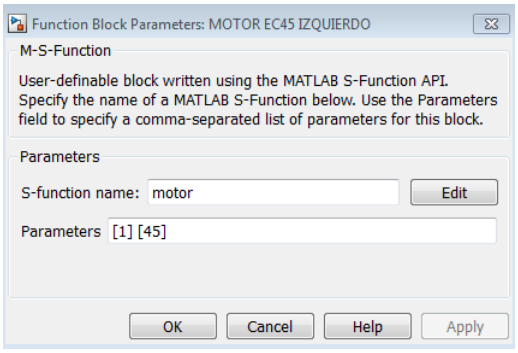


Figura 29: Parámetros de entrada del bloque motor.

Una vez acabada toda la configuración de las controladoras que se tengan en el modelo de Simulink, se procederá al movimiento de forma simultánea de todos los motores agregados. Por último, se procederá a la rutina de desconexión de forma completa llevando a la posición de origen a los motores.

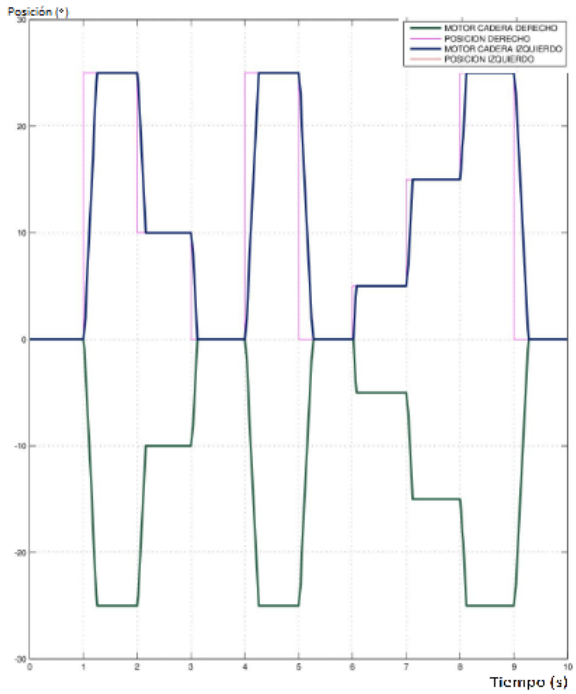


Figura 30: Movimiento de dos motores de forma simultánea.

Esta gráfica correspondiente a la figura 30 es de los motores EC45, pertenecientes a la articulación de la cadera. En ella se puede observar una respuesta simultánea de dos motores opuestos entre sí, con una respuesta totalmente simultánea entre sí. Con este ensayo se pretende demostrar la capacidad para levantarse y sentarse en trabajos futuros.

Movimiento de la cadera con carga

Se va a realizar una prueba de l articulación de la cadera para ver como responde sometido a carga. La prueba consiste en realizar una serie de movimientos poniendo y quitando una carga de 5Kg. Los resultados son los siguientes:

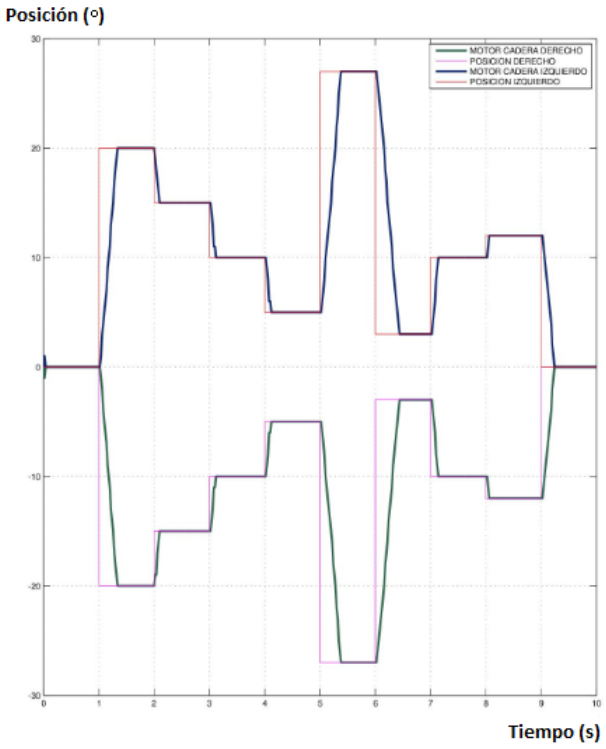


Figura 31: Movimiento de dos motores de forma simultánea con carga.

Esta gráfica correspondiente a la figura 31 es de los motores EC45 sometido a una carga de 5Kg, pertenecientes a la articulación de la cadera. En ella se puede observar una respuesta simultánea de dos motores opuestos entre sí, y a pesar de la carga el control de la posición de los motores es total. Si analizamos los resultados podemos deducir que se comporta muy similar a si estuviera trabajando en vacío, por tanto, en la cadera el control de los motores funciona de forma correcta.

Movimiento para la articulación de la rodilla

Se va a realizar una conexión, cambiando los parámetros para las rodillas, las cuales llevan incorporados unos motores EC60. Para realizar ese cambio de parámetros y como se ha dicho anteriormente, solo es necesario variar un parámetro de entrada en el bloque motor, en nuestro caso el segundo parámetro le pondremos como dato 60.

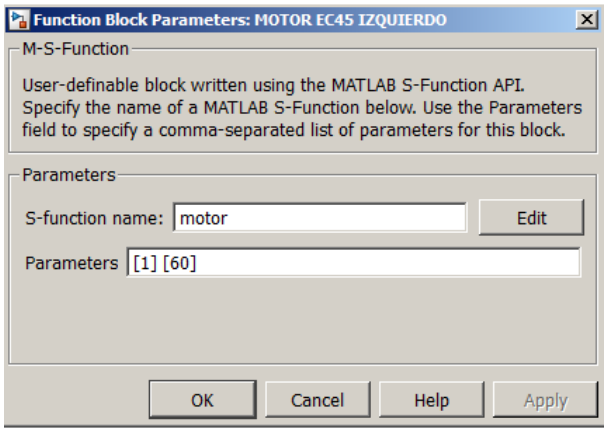


Figura 32: Parámetro de configuración del tipo de motor.

El ensayo con la articulación de las rodillas solo se puede hacer con los motores de forma simultánea, debido a una placa que une la extremidad inferior del exoesqueleto. A la hora de realizar el ensayo, se ha comprobado que no es capaz de subir por sí solo toda la estructura, sin que alguno de los dos motores entre en una situación crítica de error.

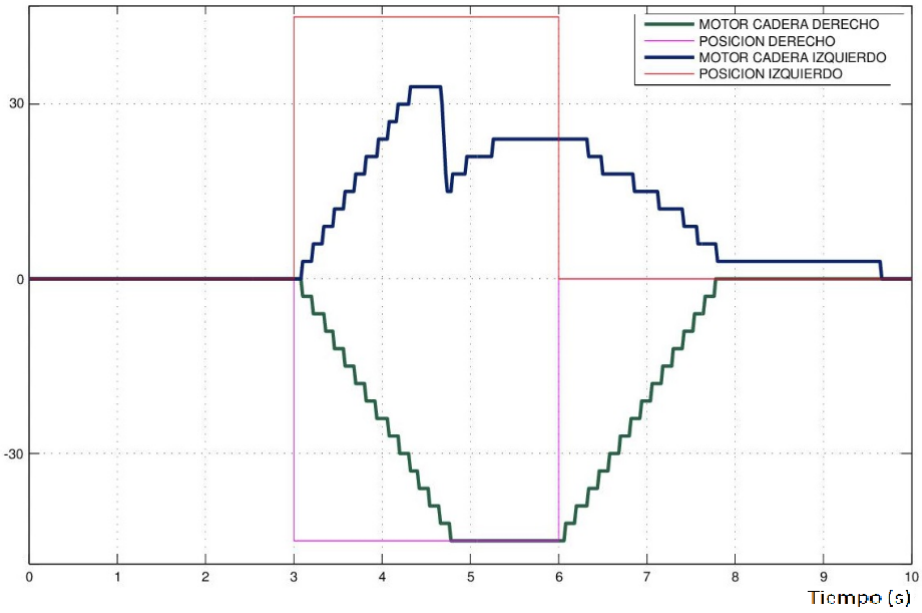


Figura 33: Motor en estado de error a 30 grados aproximadamente.

La figura 33 representa una prueba en vacío de movimiento simultáneo entre los dos motores de las rodillas. Como se puede observar, y por motivos de diseño del prototipo, estos motores o esta configuración de engranajes es incapaz de levantar la estructura, por lo tanto se está limitado a la hora de realizar los movimientos y el análisis a no excedernos de 30 grados. Como se observa en la gráfica, cuando la estructura alcanza una amplitud de 28 grados, uno de los motores falla, provocando el fallo del otro motor debido al peso. Se ha realizado una prueba en la que se levanta la estructura 45 grados con ayuda humana. Esta ayuda consiste en apoyar a los motores con una mano en la cadera para que no recaiga todo el peso sobre ellos. Como se puede observar en la prueba de levantamiento a 45 grados, la prueba ha sido satisfactoria en cuanto a simultaneidad y comportamiento de motores se refiere.

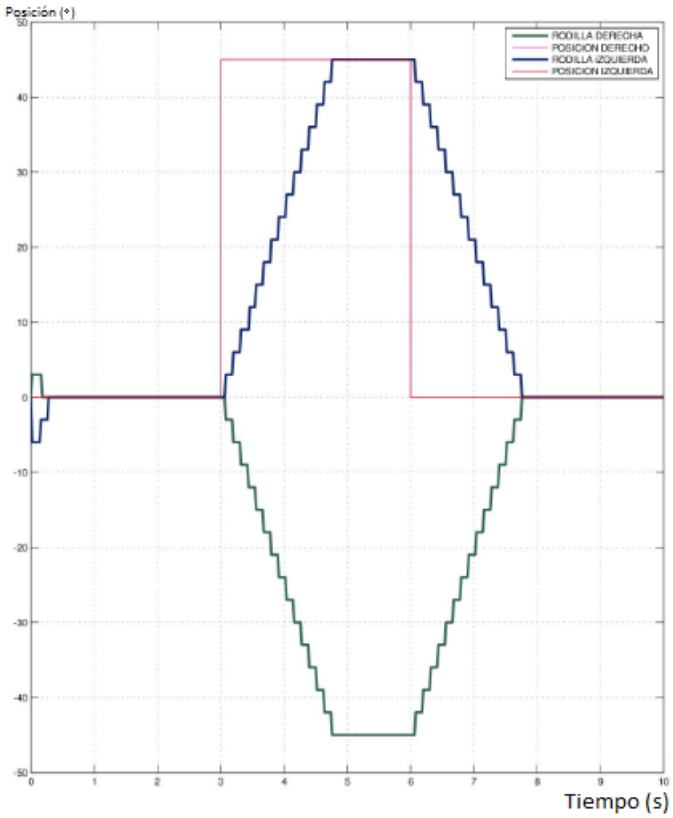


Figura 34: Movimiento de las rodillas simultáneamente.

La figura 34 representa la prueba realizada con ayuda humana del exoesqueleto para poder realizar movimientos de mas de 28^{o} en los motores de las rodillas. Como se observa, la respuesta de los motores es totalmente simultanea, de esta forma, podemos afirmar que en un futuro este algorítmo, por su eficacia, sería capaz de realizar tareas de levantarse y sentarse de forma segura.

3.5. Movimiento independiente de dos motores

En esta prueba, lo que se pretende es comprobar la seguridad en la articulación de la cadera, tanto para saber si se puede mover de forma independiente los dos lados aún estando unidos, y para ver cual es la máxima desviación que podemos tener sin que los motores den error. Para este ensayo, se va a usar el mismo esquema del apartado anterior pero con diferentes entradas de pulsos para cada motor, de esta forma se podrán variar tanto posición como la duración de ésta en cada uno de los motores.

Para realizar esta prueba, y para comprobar cual es el límite de la unión entre los dos motores de la cadera, se va a realizar un ensayo, en el cual se le van a dar diferentes posiciones a cada uno de los motores pero el ensayo se realizará en vacío.

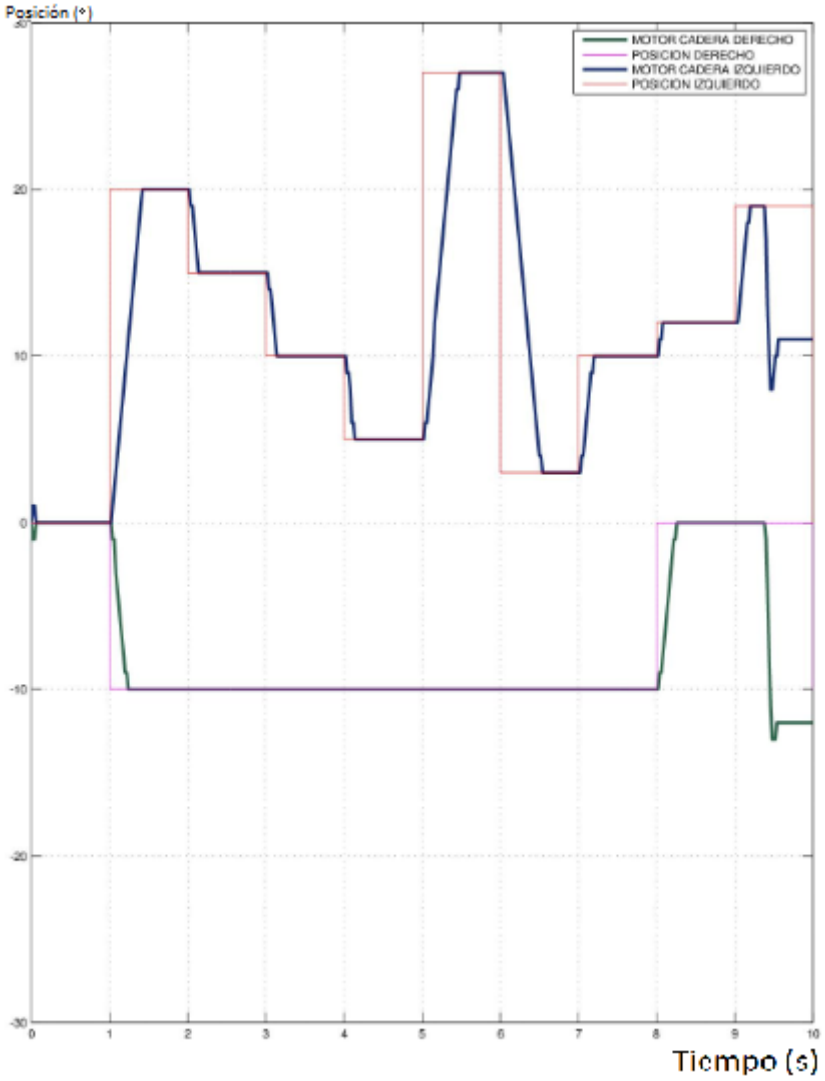


Figura 35: Desviación máxima permitida en la cadera.

Como se puede observar en la figura 35 los dos motores se mueven de forma independiente hasta que existe una desviación máxima entre ambos de 18 grados, por tanto los motores de la cadera nunca deben de estar desfasados más de 18 grados, de ser así provocaría un error que nos liberaría los motores. Esta prueba se hace ya que a la hora de caminar, la cadera de cada lado es independiente del lado opuesto, por eso se tiene que comprobar que se puede trabajar con diferentes posiciones a la vez en ambos lados de la cadera.

Para terminar el análisis de los movimientos en ambos lados de la cadera, se va a proceder a realizar una prueba en la que no se va a exceder el límite de 18 grados impuesto por la unión de la cadera. Como se puede observar en la imagen, si no se excede ese valor los motores trabajan de forma correcta.

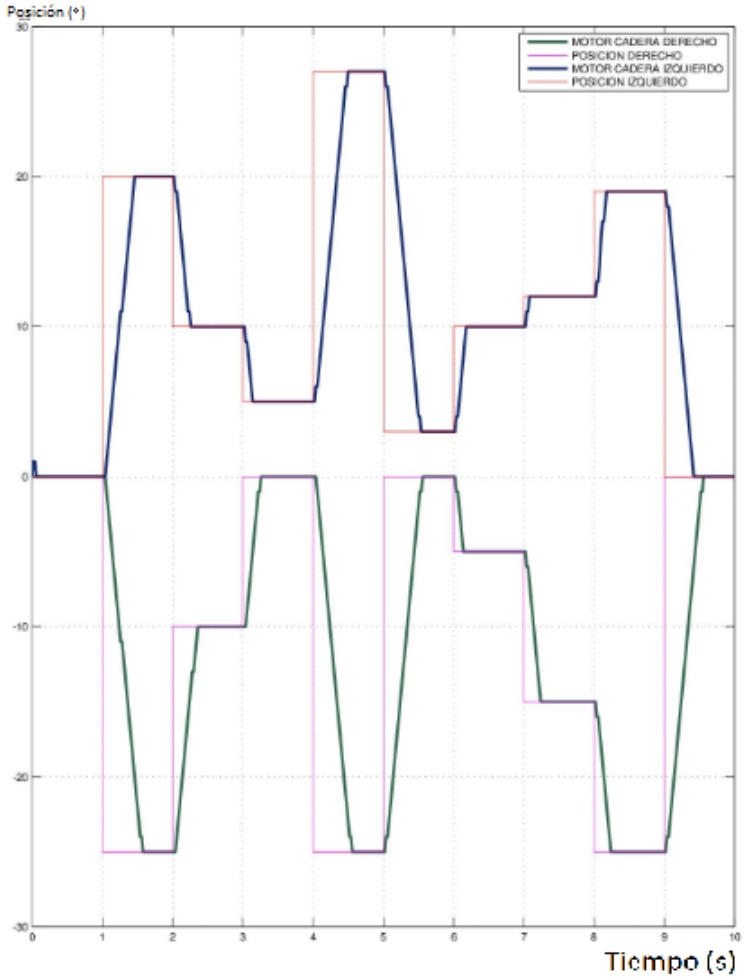


Figura 36: Movimiento independiente a ambos lados de la cadera.

La figura 36 representa la respuesta de los dos motores de la cadera, trabajando de forma simultánea pero con diferentes valores de posición. Como se puede observar la sinergia entre

ambos es total, y la respuesta a pesar de la diferencia de posiciones dadas es muy eficaz. Esta prueba se ha realizado para, en un futuro, poder realizar la tarea de caminar, ya que ésta tarea consta de movimientos simultáneos de la cadera pero con diferentes posiciones entre los diferentes lados.

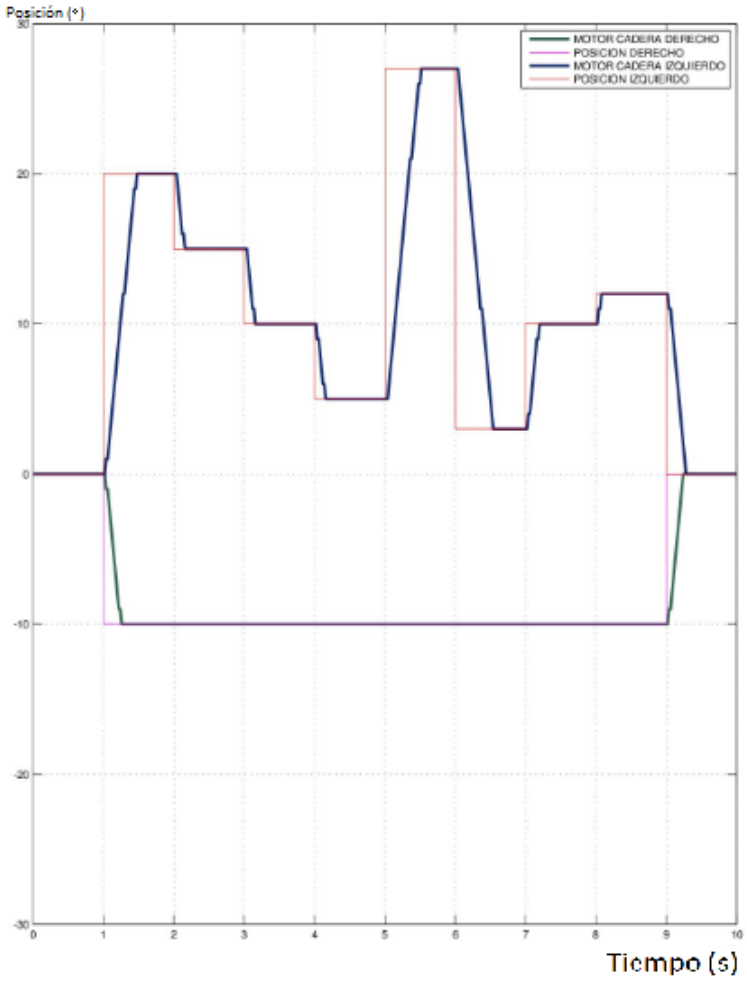


Figura 37: Movimiento independiente a ambos lados de la cadera.

La figura 37 representa el movimiento independiente de los dos motores de la cadera. En esta prueba se ha optado por darle una posición constante a un lado de la cadera mientras el otro lado va realizando cambios de posición. Como se puede observar la respuesta de ambos motores es correcta.

3.6. Movimiento de cadera y rodillas

En esta sección se va a proceder a hacer un movimiento simultáneo y controlado de las rodillas y la cadera, es decir, se va a combinar motores EC45 con motores EC60. Para realizar esto, y como se ha dicho anteriormente, en nuestro modelo de Simulink solo hay que añadir dos motores más, ya que el código realizado en Matlab permite esa flexibilidad y comodidad a la hora de realizar el modelo en Simulink, por lo tanto el modelo de Simulink es el siguiente:

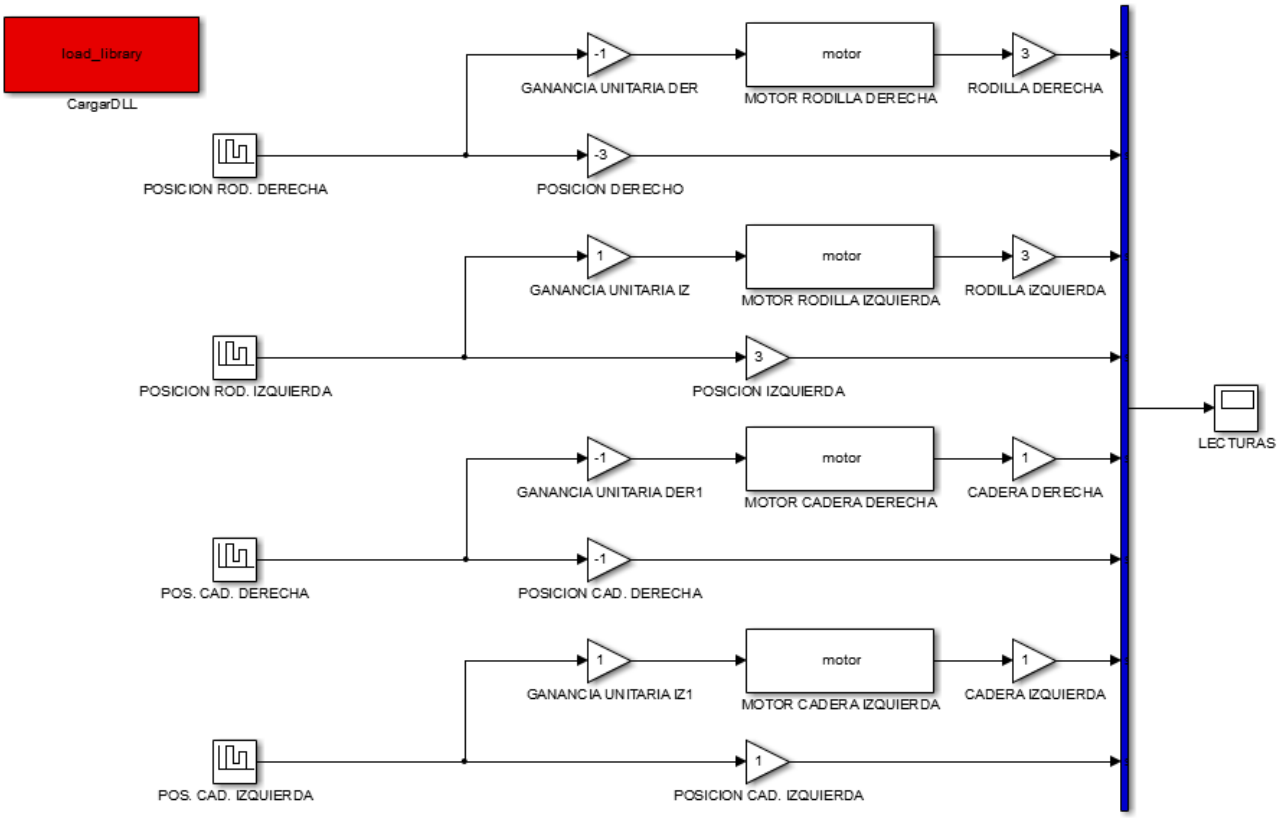


Figura 38: Modelo de Simulink para Rodillas y Cadera.

Como se puede observar en el modelo, se dispone de cuatro bloques principales llamados motor, los cuales representan a cada uno de los motores que intervienen en el movimiento: cadera izquierda, cadera derecha, rodilla izquierda y rodilla derecha. Como se ha mencionado anteriormente las pruebas de movimiento van a estar limitadas tanto en espacio para la cadera, como en límite de peso para los motores. Se van a realizar dos pruebas, una en la que se moveran las rodillas de forma simultánea, y movimiento de cadera, y la segunda consiste en la simulación de un acto de levantarse, dentro de las limitaciones de movimiento del prototipo.

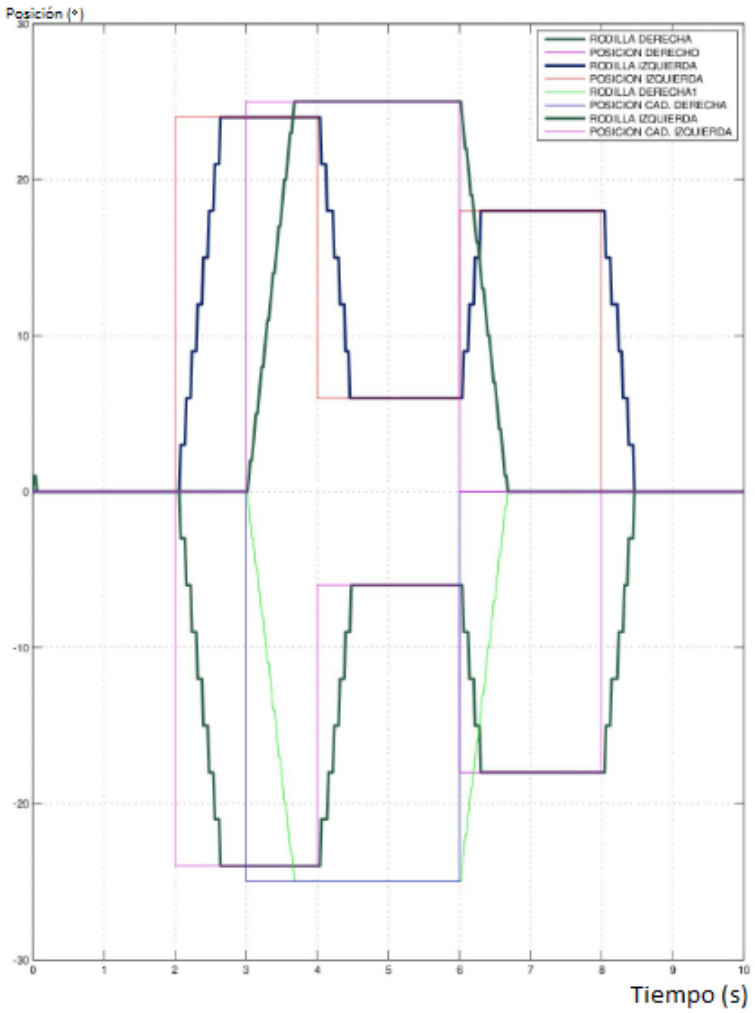


Figura 39: Movimiento de Rodillas y Cadera.

En la figura 39 se representa una prueba realizada de un movimiento simultáneo de articulaciones, pero éstas desfasadas entre sí. Como se puede observar, la simultaneidad de las articulaciones es total y la respuesta es muy eficaz. Esta prueba se ha realizado para comprobar la independencia de cada uno de los motores, así como de las articulaciones. Se observa que en ningún momento las articulaciones se ven afectadas por el movimiento de otras articulaciones.

En la figura 40 se realiza una prueba de levantarse, mantenerse en una posición, y sentarse dentro de las limitaciones de nuestro prototipo. Como se puede observar en la prueba la simultaneidad es total, ofreciendo así nuestro algorítmo una capacidad de trabajo de los motores independiente y a la vez simultáneo cuando se requiere. En la figura se observa como de forma simultánea se mueven las rodillas y la cadera hasta su posición objetivo dada. Esta trayectoria no es natural del cuerpo humano debido a las limitaciones del prototipo, pero con este ensayo se pretende demostrar una capacidad real del algorítmo para realizar este tipo de maniobras.

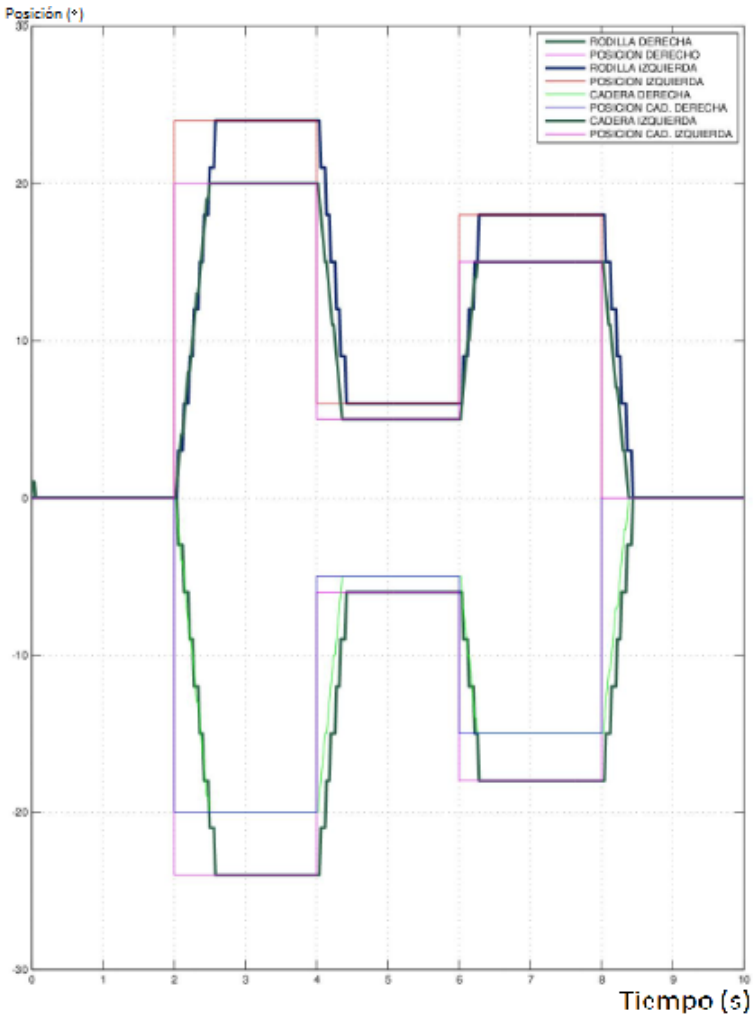


Figura 40: Comportamiento de los motores ante un acto de levantarse.

3.7. Modelo propuesto para la simulación del Exoesqueleto

Para una simulación completa del exoesqueleto se ha realizado un modelo, pero no se ha podido poner en funcionamiento debido a que un regulador de voltaje, el cual va a los motores de los tobillos está cortocircuitado internamente y no se ha remplazado. Como se puede observar en la figura, el modelo está compuesto por 6 bloques principales que funcionan de forma simultánea, de los cuales cada bloque motor pertenece a cada uno de los siguientes motores: cadera derecha, cadera izquierda, rodilla derecha, rodilla izquierda, tobillo derecho y tobillo izquierdo.

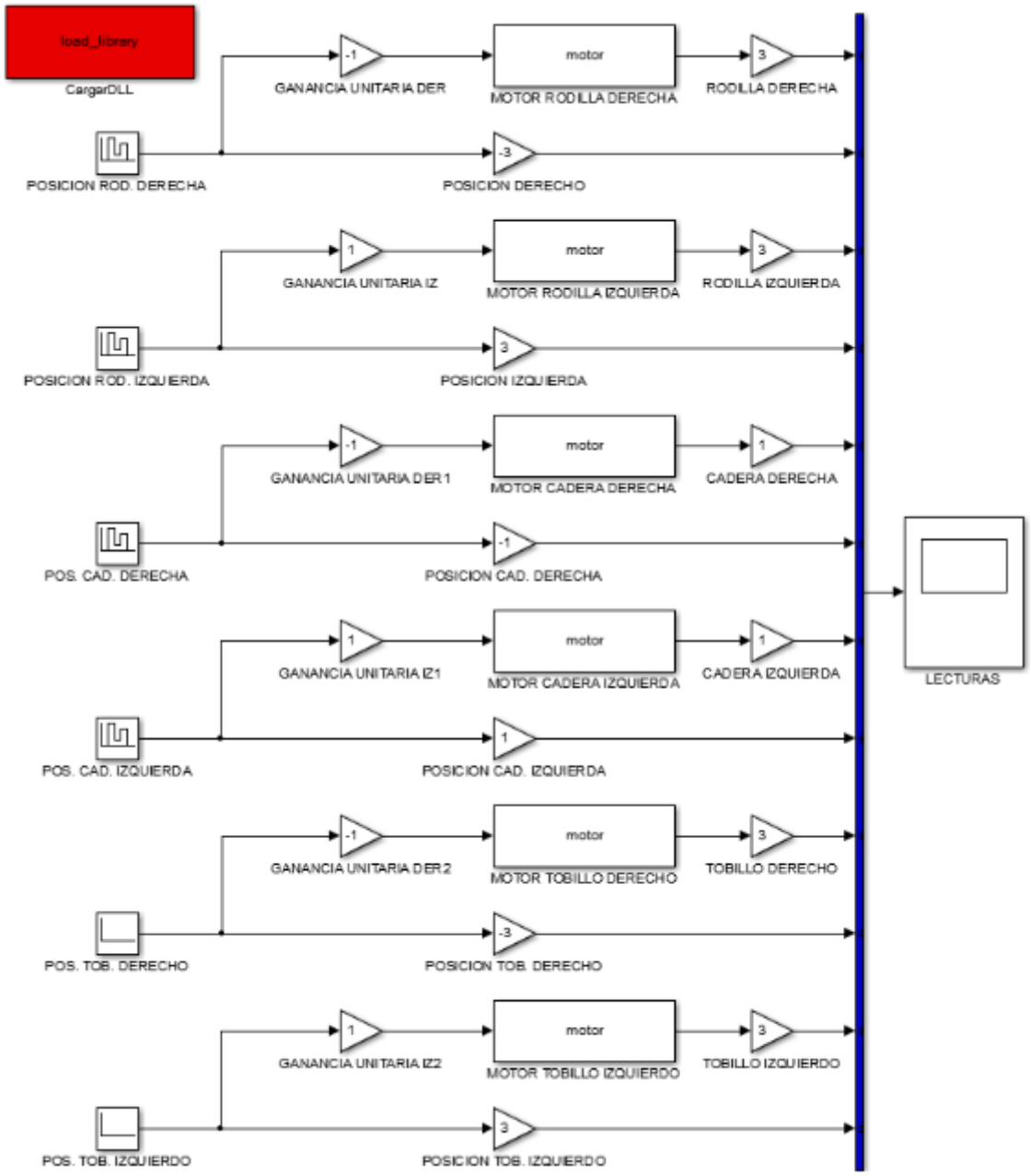


Figura 41: Modelo de Simulink del exoesqueleto.

Una vez realizado este modelo, para poner en funcionamiento y programar el exoesqueleto, se tienen que estudiar las trayectorias que se desee que realice el exoesqueleto. Para que este modelo funcione las articulaciones tienen que tener una cierta sinergia entre sí, es decir, que tienen que ser capaces de moverse de forma simultánea o independiente. Como se ha comprobado anteriormente para los diferentes ensayos, esa sinergia se puede conseguir.

Se ha propuesto este modelo debido a los resultados realizados con el modelo de las articulaciones de la cadera y la rodilla trabajando de forma simultánea. Este modelo es similar pero con una articulación más, por este motivo este modelo tiene que funcionar de la misma manera que el de dos articulaciones completas. El modelo tiene que ser capaz de hacer trabajar de forma simultánea a las articulaciones, y también tiene que ser capaz de hacerlas funcionar de forma independiente.

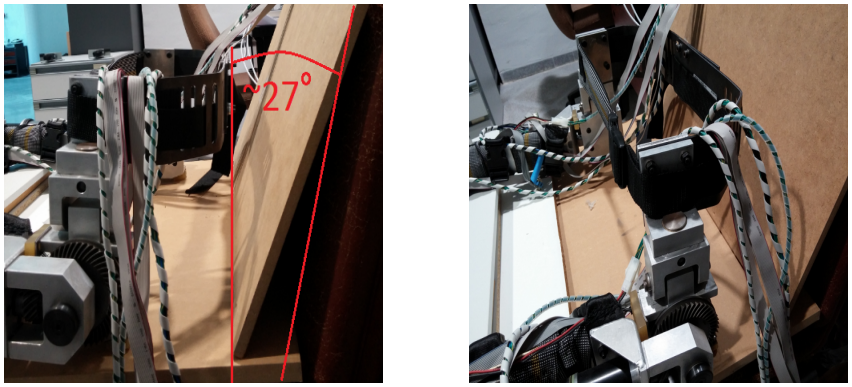
Si se analiza el diagrama de bloques propuesto se observa que cada motor es independiente del anterior, pero esto no es inconveniente para que trabajen de forma simultánea, ya que se ha comprobado anteriormente que pueden ser totalmente simultáneos trabajando independientemente. De esta forma se puede justificar la propuesta de este modelo como válido para el movimiento total del exoesqueleto.

3.8. Problemas y limitaciones

En este apartado se van a tratar todos los problemas que han surgido, y las limitaciones que se han tenido para la correcta realización del proyecto. Se han tenido limitaciones de espacio en la zona de trabajo y de movilidad, así como una limitación de movimientos por el diseño del prototipo. Por último y como problema principal, se ha tenido un problema con el derivador de voltaje que alimenta los motores de los tobillos, los cuales quedaron inutilizados y no se ha podido probar el exoesqueleto entero.

3.8.1. Limitación de espacio

Con respecto a la limitación de espacio, el prototipo se encuentra en posición sentada en una silla con respaldo inclinado. Al estar el prototipo sentado y no tener ningún tipo de medio para poder realizar movimientos sin riesgo de seguridad, solamente se podrá mover mientras esté sentado. En esta postura y como se muestra en la imagen, solo se dispone de aproximadamente 27 grados de movimiento, por lo tanto es un movimiento bastante limitado, pero suficiente para realizar diversas pruebas configuraciones.



(a) Limitación respaldo (b) Limitación cableado

Figura 42: Espacio disponible para realizar pruebas.

Otra limitación relacionada con el espacio, es el cableado, que se situa de forma libre entre el exoesqueleto y las controladoras. Este cableado es un riesgo de seguridad, debido a que si en algún momento uno de los cables se queda enganchado en algún sitio, se puede producir la rotura del mismo o un error en la controladora, el cual nos puede provocar un importante daño en nuestro prototipo si nos encontramos en ese momento trabajando con el y en movimiento.

3.8.2. Limitación de peso

Como se ha comprobado cuando se ha intentado realizar movimientos con la articulación de la rodilla, los motores seleccionados no son capaces de aportar toda el par necesario para mover el prototipo. Se ha observado que debido al peso del prototipo y que tiene que soportar cuando realiza el movimiento para levantarse, cuando llega a una cierta inclinación, los motores entran en corte. Si los motores entran en corte, automáticamente aparece un error interno de la controladora por el cual deja libre a los motores, y en un caso real, la persona a la que estuviera ayudando le produciría una caida.

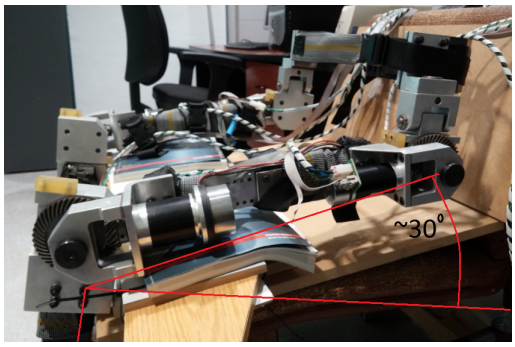


Figura 43: Limitación de carga de los motores de la rodilla.

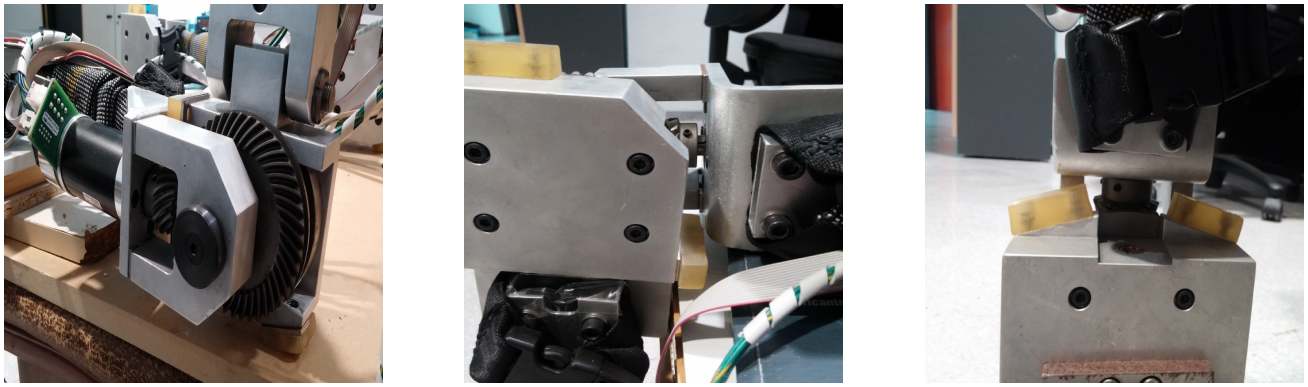
3.8.3. Limitación de movimientos naturales, topes

Para realizar los movimientos de levantarse, sentarse, andar, ... Se necesita una disponibilidad de capacidad de movimiento del prototipo muy amplia para la realización de una forma natural de todos estos movimientos, la cual este prototipo no cumple de forma natural a ciertos movimientos.

Para el movimiento de levantarse y sentarse, se necesita desplazar el centro de gravedad de el peso del cuerpo para mantener el equilibrio, y esto se consigue inclinando la cadera hacia delante mientras estamos sentados. Como se puede observar en la imagen de la limitación de cadera, ese movimiento no lo permite debido al diseño del prototipo, por lo tanto con este prototipo sería imposible realizar una acción natural tanto de levantarse como de sentarse.

Otras limitaciones importantes de movimiento que se encuentran están en las articulaciones de la rodilla y de los tobillos. Como se puede observar en las imágenes, los rangos de movimientos son amplios para los movimientos normales, pero si queremos realizar algún movimiento más

complejo o diferente, este prototipo nos impediría poder realizar dichos movimientos debido a su diseño mecánico.



(a) Limitación Cadera (b) Limitación Rodilla (c) Limitacion Tobillo

Figura 44: Limitación de movimientos del prototipo.

3.8.4. Problema del Regulador de tensión para los motores EC60

Se ha tenido un problema importante, el cual todavía no se ha resuelto, y es la rotura de un regulador de voltaje, el cual alimenta a los motores EC60 asociado en un primer momento con el lado izquierdo del exoesqueleto. El problema es aparentemente un cortocircuito, ya que se han realizado unas pruebas superficiales para ver que regulador o cuales reguladores estaban fallando y para detectar el porqué del fallo. Una vez detectado el regulador se ha anulado para no provocar más fallos y se ha modificado el esquema de conexión para poder aprovechar el otro regulador disponible. Para poder avanzar en el proyecto se ha conectado la rodilla izquierda al regulador de voltaje que alimenta a los motores EC60 del lado derecho, prescindiendo del tobillo derecho.

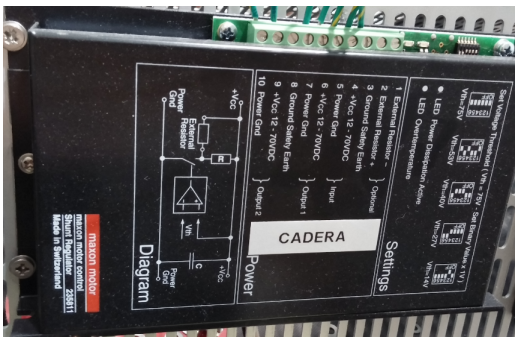


Figura 45: Regulador averiado.

3.9. Mejoras propuestas

Para intentar solucionar estas limitaciones y problemas, y para optimizar todo el proceso se proponen una serie de mejoras, las cuales engloban a aspectos como la seguridad para trabajar, otras opciones de comunicación posiblemente más efectivas, ampliación de sensores para una mayor seguridad y un nuevo diseño mecánico que permita unos movimientos más naturales.

3.9.1. Sistemas de seguridad

Para la mejora del aspecto de seguridad, se propone un banco de pruebas en el que se pueda trabajar con el exoesqueleto y realizar pruebas sin ningún tipo de limitación. Para esta mejora se propone que el exoesqueleto se sitúe colgado sobre un polipasto. Esta acción permite poder simular cualquier tipo de movimiento sin riesgo de peligro y con total seguridad.

3.9.2. Protocolo de comunicación

Actualmente se dispone de una comunicación vía USB, independiente para cada controladora. Esta comunicación presenta varios inconvenientes, entre los que se encuentra la cantidad de conexiones que hay que realizar con el ordenador, y por tanto más probabilidad de que algún puerto USB falle.

Para optimizar lo realizado, se propone una comunicación mediante una red de sensores CAN. Estas redes son más factibles debido a que solo se haría una conexión al ordenador, el cual enviaría las ordenes a los diferentes sensores que están conectados entre sí, pero cada uno es un nodo diferente, pudiendo de esta forma poder intercambiar datos entre ellos y evitar conexiones independientes de cada controladora por separado. De esta forma se dispondría de una red interconectada entre sí y capaz de interactuar ante cualquier fallo.

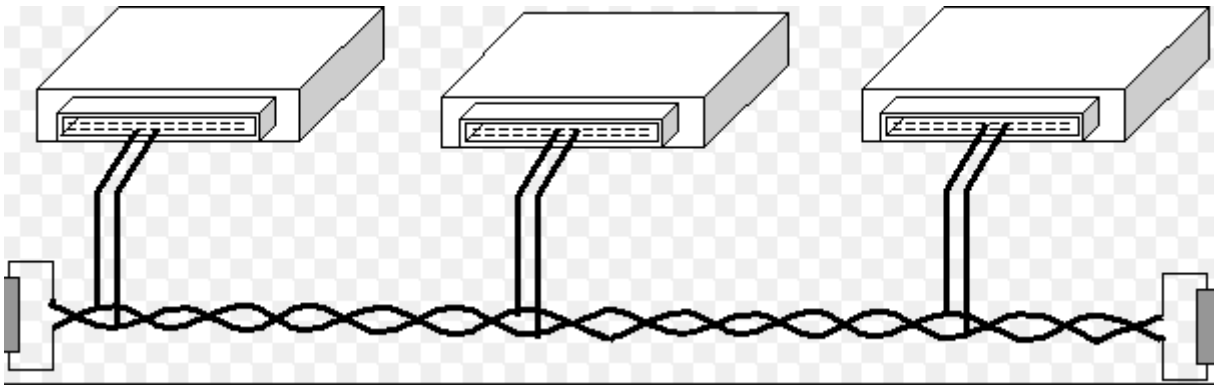


Figura 46: Red de comunicación, protocolo CANOpen.

3.9.3. Implantación de Encoders Absolutos

Actualmente se tiene un problema a la hora de la configuración de los motores, debido a que una vez que se apaga la alimentación, todos los encoders establecen su nuevo 0 cuando se le alimenta, de esta forma es imposible saber en la posición en la que está el exoesqueleto al iniciar.

Para solucionar este problema se propone implantar algún sistema de seguridad que nos permita saber la posición del exoesqueleto de forma exacta. Para ello se propone la implantación de una red de encoders absolutos, los cuales nos permitan saber en cada momento en que posición estamos, para de esta forma configurar los motores y las trayectorias de forma correcta y segura antes de cada movimiento. Se proponen los encoders absolutos de la marca Maxon, ya que son perfectamente compatibles con las entradas de los controladores utilizados.

3.9.4. Implantación de Giroscopio

A la hora de realizar un control de equilibrio en suelos irregulares se observa que no se dispone de medios electrónicos para saber en cada momento el grado de equilibrio que tenemos. Para solucionar esto se propone la implantación de un giroscopio en la cadera para controlar la nivelación de la persona en cada momento. De esta forma, cuando el individuo pase por suelos irregulares, será capaz de controlar el equilibrio y la posición de sus articulaciones en función de los datos obtenidos por el giroscopio.

3.9.5. Diseño del prototipo

Para solucionar los problemas de diseño de prototipo se proponen diferentes medidas: reducir el peso con otro material y reducir ángulos entre las extremidades del exoesqueleto.

Para la reducción de peso del exoesqueleto, se propone la utilización de otro tipo de material más ligero, pero capaz de resistir la carga para la que el exoesqueleto se necesita. otra medida para la reducción de peso es la de la optimización del diseño evitando sobredimensionados de la estructura del exoesqueleto.

Figura 47: Posible mejora de diseño.

Para poder aumentar el rango de movimientos del exoesqueleto, se propone un diseño con una unión entre extremidades circular, lo cual, nos permite unos movimientos prácticamente ilimitados. Como el cuerpo humano tiene unas limitaciones, se propone poner unos sistemas de seguridad en el límite de los movimientos naturales, estas medidas de protección podrían ser unos interruptores finales de carrera, encoders, o cualquier otro sistema mecánico o electrónico capaz de ofrecer la máxima seguridad.

3.10. Trabajos futuros respecto a algorítmos de control

Como continuación y finalización del proyecto se proponen trabajos futuros con respecto a algorítmos de control de trayectorias. Si se analiza el diagrama de bloques propuesto se observa que está preparado para sustituir las entradas a los motores por una función que genere trayectorias. Esta función tiene que ser capaz de comunicar todos los motores y condicionar las entradas de cada uno con respecto a la trayectoria que se vaya a realizar.

Trayectoria de levantarse y sentarse

Para la trayectoria de levantarse y sentarse, se ha tomado como referencia el trabajo de Karlj y Bajd (1989) donde se describe que la acción de ponerse en pie y sentarse está dividida en fases, las cuales se pueden representar en la siguiente imagen.

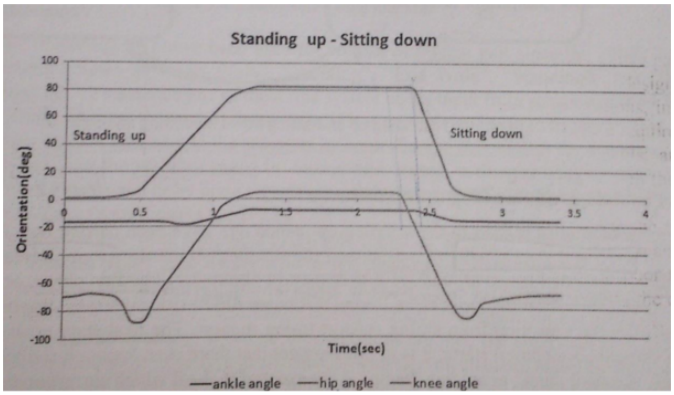


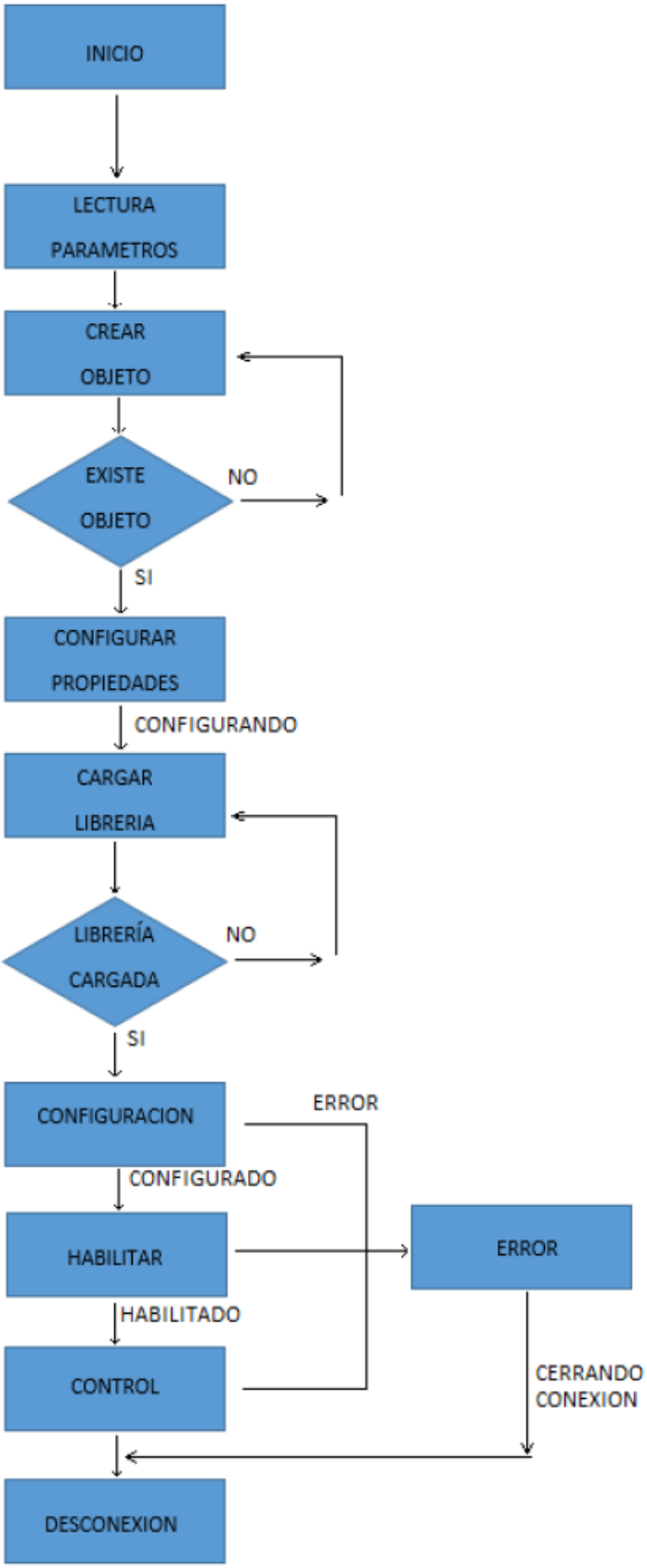
Figura 48: Trayectoria de levantarse y sentarse, Karlj y Bajd (1989).

Como se observa en la gráfica, para levantarse se necesitan realizar diferentes movimientos. En primer lugar se tiene que inclinar la cadera hacia adelante para conseguir desplazar el centro de gravedad, en segundo lugar, se eleva el cuerpo, moviendo todas las articulaciones de forma simultánea para no perder el equilibrio, y en último lugar se procede a la estabilización de la posición totalmente erguido.
Para realizar la trayectoria de sentarse hay que realizar de forma rigurosa pero a la inversa, una trayectoria similar, cuidando en todo momento el equilibrio y el centro de gravedad para no producir una caída.

Esta trayectoria sería un ejemplo de como afrontar la parte del algorítmo de control de trayectorias. Las trayectorias principales a estudiar serían: Ponerse de pie y sentarse, andar, subir y bajar escaleras, mantener el equilibrio en plataformas oscilantes, etc.

4. Anexos

4.1. Flujograma de funcionamiento



4.2. Algorítmo de programación en Matlab

A continuación se muestra un ejemplo completo de una rutina de movimiento. Como se puede comprobar está todo en el mismo orden anteriormente explicado, de esta forma se expone un código con el que realizar de forma satisfactoria un movimiento de un motor. Este código está referenciado al movimiento de la cadera, si se quisiera otra extremidad, ya que cambiaría el motor, solamente habría que cambiar la clase donde estén las propiedades con los valores correctos de ese motor.

```
%---RUTINA PARA TESTEAR UN MOTOR CONECTADO A USB0, ESTA RUTINA ESTA CONFIGURADA PARA UN
%MOTOR EC45, SI QUEREMOS UN MOTOR EC60 CAMBIAREMOS LA PRIMERA LINEA DE
%CODIGO POR LA SEGUNDA.-----------------------------------------------

%--------SE CREA EL OBJETO CON EL QUE VAMOS A TRABAJAR-----------------
epos = class_epos();    %MOTOR EC45
%epos = class_epos60(); %MOTOR EC60
epos.PortName='USB0';
cerrar=1;
%---------CARGAR LIBRERIA-------------------------------------------------
[notfound, warnings]= loadlibrary ('EposCmd', 'Definitions.h');

%--------CONFIGURACION Y HABILITACION DEL HADWARE-----------------------
%--------PARAMETRIZACIÓN DE LA CONEXIÓN----------------------
[epos.KeyHandle, epos.DeviceName, epos.ProtocolStackName, epos.InterfaceName, epos.PortNam
 epos.ErrorCode] = calllib ('EposCmd', 'VCS_OpenDevice',epos.DeviceName, epos.ProtocolStac
 epos.InterfaceName, epos.PortName, epos.ErrorCode);

[epos.ReturnValue, epos.KeyHandle, epos.ErrorCode] = calllib ('EposCmd', 'VCS_SetProtocolSt
epos.KeyHandle, epos.Baudrate,epos.Timeout , epos.ErrorCode);

[epos.ReturnValue, epos.KeyHandle, epos.ErrorCode] = calllib ('EposCmd', 'VCS_ClearFault',
epos.KeyHandle,epos.NodeId, epos.ErrorCode);

[epos.ReturnValue,epos.KeyHandle, epos.ErrorCode] = calllib ('EposCmd', 'VCS_SetPositionRe
 epos.KeyHandle,epos.NodeId, epos.P, epos.I, epos.D, epos.ErrorCode);

[epos.ReturnValue,epos.KeyHandle, epos.ErrorCode] = calllib ('EposCmd', 'VCS_SetVelocityUn
 epos.KeyHandle,epos.NodeId, epos.VelDimension, epos.VelNotation, epos.ErrorCode);
```

```
%--------CONFIGURACION DE MOTORES, SENSORES Y PARAMETROS LIMITE-----------------------
[epos.ReturnValue,epos.KeyHandle, epos.ErrorCode] = calllib ('EposCmd', 'VCS_SetMotorType'
 epos.KeyHandle,epos.NodeId, epos.MotorType,  epos.ErrorCode);

[epos.ReturnValue,epos.KeyHandle, epos.ErrorCode] = calllib ('EposCmd', 'VCS_SetEcMotorPar
 epos.KeyHandle,epos.NodeId, epos.NominalCurrent,epos.MaxOutputCurrent,epos.ThermalTimeCon
 epos.NbOfPolePairs,epos.ErrorCode);

[epos.ReturnValue,epos.KeyHandle, epos.ErrorCode] = calllib ('EposCmd', 'VCS_SetSensorType
 epos.KeyHandle,epos.NodeId, epos.SensorType, epos.ErrorCode);

[epos.ReturnValue,epos.KeyHandle, epos.ErrorCode] = calllib ('EposCmd', 'VCS_SetIncEncoder
 epos.KeyHandle,epos.NodeId, epos.EncoderResolution, epos.InvertedPolarity, epos.ErrorCode

[epos.ReturnValue,epos.KeyHandle, epos.ErrorCode] = calllib ('EposCmd', 'VCS_SetMaxAcceler
 epos.KeyHandle,epos.NodeId, epos.MaxAcceleration, epos.ErrorCode);

[epos.ReturnValue,epos.KeyHandle, epos.ErrorCode] = calllib ('EposCmd', 'VCS_SetMaxFollowi
 epos.KeyHandle,epos.NodeId, epos.MaxFollowingError, epos.ErrorCode);

[epos.ReturnValue,epos.KeyHandle, epos.ErrorCode] = calllib ('EposCmd', 'VCS_SetMaxProfile
 epos.KeyHandle, epos.NodeId, epos.MaxProfileVelocity, epos.ErrorCode);


[epos.ReturnValue,epos.KeyHandle, epos.ErrorCode] = calllib ('EposCmd', 'VCS_SetOperationM
 epos.KeyHandle,epos.NodeId, epos.Mode, epos.ErrorCode);

%--------HABILITAR CONTROLADORA-----------------------
[epos.ReturnValue,epos.KeyHandle, epos.ErrorCode] = calllib ('EposCmd', 'VCS_SetEnableStat
 epos.KeyHandle, epos.NodeId, epos.ErrorCode);

disp('EPOS2 Habilitada, Puede comenzar a testear...');

%-----------------RUTINA DE OPERACION-----------------------
%--------ESTABLECER MODO DE OPERACION-----------------------
[epos.ReturnValue,epos.KeyHandle, epos.ErrorCode] = calllib ('EposCmd', 'VCS_ActivateProfi
 epos.KeyHandle, epos.NodeId, epos.ErrorCode)

%--------MOVIMIENTO DE ARTICULACIONES-----------------------
[epos.ReturnValue,epos.KeyHandle, epos.ErrorCode] = calllib ('EposCmd', 'VCS_SetPositionPr
```

```
 epos.KeyHandle, epos.NodeId, epos.ProfileVelocity, epos.ProfileAcceleration,...
 epos.ProfileDeceleration, epos.ErrorCode);

%TIPO DE PERFIL, 0--> TRAPEZOIDAL, 1--> SENOIDAL
profile = 1;
number = 0;

[epos.ReturnValue,epos.KeyHandle, profile, number ,epos.ErrorCode] = calllib ('EposCmd',..
 'VCS_SetObject', epos.KeyHandle, epos.NodeId, 24710, 0, profile, 2, number, epos.ErrorCod

[epos.ReturnValue, epos.KeyHandle, epos.ErrorCode] = calllib ('EposCmd', 'VCS_MoveToPositi
 epos.KeyHandle, epos.NodeId, pasos, epos.Absolute, epos.Inmediately, epos.ErrorCode);
%--------LECTURA DEL SENSOR DE POSICION-----------------------
[epos.ReturnValue, epos.KeyHandle, epos.PositionIs, epos.ErrorCode] = calllib ('EposCmd',.
 'VCS_GetPositionIs', epos.KeyHandle, epos.NodeId, epos.PositionIs, epos.ErrorCode);

%------------RUTINA DE DESCONEXION----------------------
if cerrar==1

%--------DESHABILITAR CONTROLADORA----------------------
[epos.ReturnValue,epos.KeyHandle, epos.ErrorCode] = calllib ('EposCmd', 'VCS_SetDisableSta
 epos.KeyHandle,epos.NodeId, epos.ErrorCode);

%--------DESCONEXION DE DISPOSITIVOS----------------------
[epos.ReturnValue,epos.KeyHandle, epos.ErrorCode] = calllib ('EposCmd', 'VCS_CloseDevice',
 epos.KeyHandle, epos.ErrorCode);

[epos.ReturnValue, epos.ErrorCode] = calllib ('EposCmd', 'VCS_CloseAllDevices', epos.Error

%--------QUITAR LIBRERIA DE MEMORIA----------------------
unloadlibrary EposCmd

end
```

4.3. Algorítmo de programación en Simulink

Como ejemplos de programación de simulink se van a poner las páginas de código correspondientes a los bloques motor y load library. En estos bloques se realiza toda la acción de control y comunicación.

Script para cargar librería, Load Library.m

```
function load_library(block)

setup(block);

function setup(block)

% Register number of ports
block.NumInputPorts  = 0;
block.NumOutputPorts = 0;

global gEpos;
global gLibraryLoad;
global gEpos2Conected;

% Register parameters
block.NumDialogPrms     = 0;
block.SampleTimes = [0 0];
block.SimStateCompliance = 'DefaultSimState';
block.RegBlockMethod('PostPropagationSetup',    @DoPostPropSetup);
block.RegBlockMethod('InitializeConditions', @InitializeConditions);
block.RegBlockMethod('Start', @Start);
block.RegBlockMethod('Outputs', @Outputs);     % Required
block.RegBlockMethod('Update', @Update);
block.RegBlockMethod('Derivatives', @Derivatives);
block.RegBlockMethod('Terminate', @Terminate); % Required

function DoPostPropSetup(block)
block.NumDworks = 1;

  block.Dwork(1).Name            = 'x1';
  block.Dwork(1).Dimensions      = 1;
  block.Dwork(1).DatatypeID      = 0;      % double
```

```
  block.Dwork(1).Complexity      = 'Real'; % real
  block.Dwork(1).UsedAsDiscState = true;


function InitializeConditions(block)
    global gEpos;

    if ~iscell(gEpos)
        gEpos = cell(1);
        disp('ya es cell por Conf');
    end   %Fin de inicializar condiciones

function Start(block)
global gLibraryLoad;
set_param('exoconencoderintegrator/CargarDLL','BackgroundColor','white')
[notfound, warnings]= loadlibrary ('EposCmd', 'Definitions.h');
gLibraryLoad = libisloaded('EposCmd')+1;
set_param('exoconencoderintegrator/CargarDLL','BackgroundColor','green')
block.Dwork(1).Data = 0;

%OUTPUTS
function Outputs(block)
block.Dwork(1).Data;

function Update(block)

block.Dwork(1).Data = 1;

function Derivatives(block)

function Terminate(block)
pause(2);

unloadlibrary EposCmd;
set_param('exoconencoderintegrator/CargarDLL','BackgroundColor','red')
```

Script que realiza todo la tarea de configuración y control, Motor.m

```
function motor(block)

setup(block);

function setup(block)

% Register number of ports
    block.NumInputPorts  = 1;
    block.NumOutputPorts = 1;
    %global gKeyHandle;
    %block.SetPreCompOutPortInfoToDynamic;

% Override output port properties
    block.OutputPort(1).Dimensions = 1;
    block.OutputPort(1).DatatypeID  = 0; % double
    block.OutputPort(1).SamplingMode = 'sample';
    block.OutputPort(1).Complexity  = 'Real';

% Override input port properties
    block.InputPort(1).Dimensions   = 1;
    block.InputPort(1).DatatypeID  = 0; % double
    block.InputPort(1).SamplingMode = 'sample';
    block.InputPort(1).Complexity  = 'Real';

% Register parameters
    block.NumDialogPrms     = 2;

    block.SampleTimes = [0 0];

    block.SimStateCompliance = 'DefaultSimState';

    block.RegBlockMethod('PostPropagationSetup',    @DoPostPropSetup);
    block.RegBlockMethod('InitializeConditions', @InitializeConditions);
    block.RegBlockMethod('Start', @Start);
    block.RegBlockMethod('Outputs', @Outputs);     % Required
    block.RegBlockMethod('Update', @Update);
    block.RegBlockMethod('Derivatives', @Derivatives);
    block.RegBlockMethod('Terminate', @Terminate); % Required
```

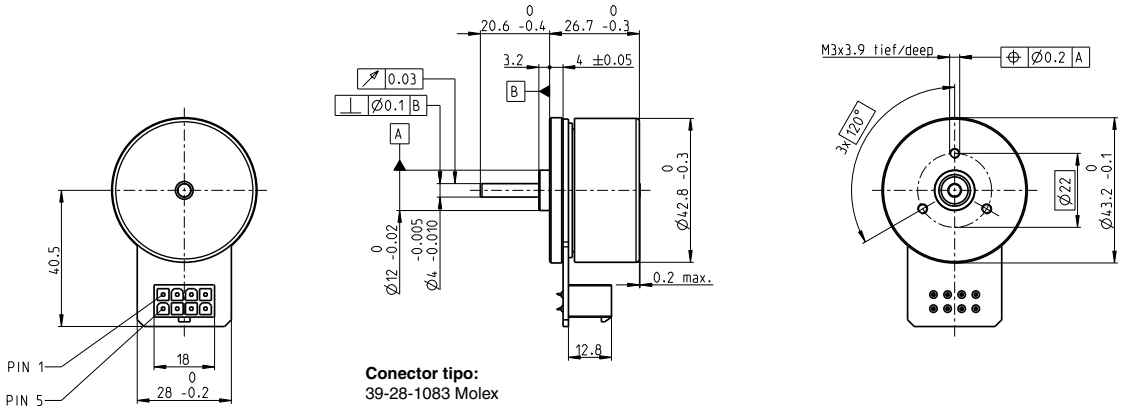
```
function DoPostPropSetup(block)
    block.NumDworks = 1;
    block.Dwork(1).Name            = 'x1';
    block.Dwork(1).Dimensions      = 1;
    block.Dwork(1).DatatypeID      = 0;      % double
    block.Dwork(1).Complexity      = 'Real'; % real
    block.Dwork(1).UsedAsDiscState = true;

function InitializeConditions(block)

global gEpos

%%COMPROBAMOS SI ES EL EC45 O EC60 PARA CREAR SU OBJETO CORRESPONDIENTE
    if block.DialogPrm(2).Data==45
        epos=class_epos;
        epos.PortName =block.DialogPrm(1).Data;
        epos.PortName = USBPortName(epos.PortName);
        %epos.NodeId = block.DialogPrm(1).Data+1; NO PUEDO CAMBIARLE EL ID
        %PORQUE SE BLOQUEA LA SIMULACION
    end

    if block.DialogPrm(2).Data==60
        epos=class_epos60;
        epos.PortName =block.DialogPrm(1).Data;
        epos.PortName = USBPortName(epos.PortName);
        epos.NodeId = block.DialogPrm(1).Data+1;
    end
%% FIN DE LA ASIGNACION DE MOTOR EC45 O EC60

if ~iscell(gEpos)
    gEpos = cell(1);
    disp('ya es cell por READ');
end

    gEpos{block.DialogPrm(1).Data+1} = epos;
    gEpos{block.DialogPrm(1).Data+1}

function Start(block)
```

```
function Outputs(block)
global gEpos;
global gLibraryLoad;

if (gEpos{block.DialogPrm(1).Data+1}.Connected==1)
    library_epos ('get_position_is', '', gEpos{block.DialogPrm(1).Data+1});
    % OBTENER POSICION DEL SENSOR


        if(gEpos{block.DialogPrm(1).Data+1}.PositionIs ~= block.InputPort(1).Data)
        %fprintf('\n(%s, %i, %i)',gEpos{block.DialogPrm(1).Data+1}.PortName,...
        %gEpos{block.DialogPrm(1).Data+1}.TargetPosition,
             gEpos{block.DialogPrm(1).Data+1}.PositionIs); %MOSTRAR POSICION
        gEpos{block.DialogPrm(1).Data+1}.TargetPosition = block.InputPort(1).Data;
         % NUMERO DE PASOS O POSICION ABSOLUTA
        movements ('move_to_position', 'Moviendo',  gEpos{block.DialogPrm(1).Data+1});
         % MOVER EL MOTOR
        %pause(0.5);
        movements ('get_position_is', '', gEpos{block.DialogPrm(1).Data+1});
          % OBTENER POSICION DEL SENSOR

        %EL GET POSITION LO DIVIDIMOS ENTRE 1000 PARA PASAR DE PASOS A
        %GRADOS
        fprintf('\t(%s, %i, %i)',gEpos{block.DialogPrm(1).Data+1}.PortName,...
            gEpos{block.DialogPrm(1).Data+1}.TargetPosition,
             (gEpos{block.DialogPrm(1).Data+1}.PositionIs/904.53)); %MOSTRAR POSICION DEL
        end



    elseif (gLibraryLoad==2) %&& gEpos{block.DialogPrm(1).Data+1}.TargetPosition==0)
%disp('CARGADA');
library_epos ('connect_device_manually', 'Conectando...',  gEpos{block.DialogPrm(1).Data+1
%operating_mode(gEpos{block.DialogPrm(1).Data+1}.Mode, 'Seleccionando modo de operacion',
      gEpos{block.DialogPrm(1).Data+1});
%library_epos ('enable', 'Habilitando...',  gEpos{block.DialogPrm(1).Data+1});
library_epos ('configuration_devices', 'Configurando motores y sensores...',
     gEpos{block.DialogPrm(1).Data+1});
library_epos ('enable', 'Habilitando...',  gEpos{block.DialogPrm(1).Data+1});
```

```
operating_mode(gEpos{block.DialogPrm(1).Data+1}.Mode, 'Seleccionando modo de operacion',
     gEpos{block.DialogPrm(1).Data+1});

    if(gEpos{block.DialogPrm(1).Data+1}.ReturnValue~=0)
        gEpos{block.DialogPrm(1).Data+1}.Connected=1;
        disp('CORRECTO');
    else
        disp('ERRORR');
    end
end

  % DIVIDIMOS ENTRE 904.53 PARA PASAR DE PASOS A GRADOS
block.OutputPort(1).Data=double((gEpos{block.DialogPrm(1).Data+1}.PositionIs/904.53));


function Update(block)

function Derivatives(block)

function Terminate(block)
global gEpos;
 gEpos{block.DialogPrm(1).Data+1}.TargetPosition = 0;

% MOVER EL MOTOR
   movements ('move_to_position', 'Moviendo al origen por movements',
              gEpos{block.DialogPrm(1).Data+1});
   library_epos ('disable', 'Deshabilitando...', gEpos{block.DialogPrm(1).Data+1});
```

4.4. Hojas de características Maxon

EC 45 flat ∅42.8 mm, Conmutación electrónica (Brushless), 70 W



Conector tipo:
39-28-1083 Molex

M 1:2

Programa Stock
Programa Estándar
Programa Especial (previo encargo)

	Referencia			
con sensores Hall	397172	402685	402686	402687
Datos del motor (provisionales)				
Valores a tensión nominal				
1 Tensión nominal V	24	30	36	48
2 Velocidad en vacío rpm	6110	6230	6330	3440
3 Corriente en vacío mA	234	194	166	48.1
4 Velocidad nominal rpm	4860	4990	5080	2540
5 Par nominal (máx. par en continuo) mNm	128	112	108	134
6 Corriente nominal (máx. corriente en continuo) A	3.21	2.36	1.93	0.936
7 Par de arranque mNm	1460	1170	1100	915
8 Corriente de arranque A	39.5	25.8	20.7	6.97
9 Máx. rendimiento %	85	84	83	84
Características				
10 Resistencia en bornes fase-fase Ω	0.608	1.16	1.74	6.89
11 Inductancia en bornes fase-fase mH	0.463	0.691	0.966	5.85
12 Constante de par mNm / A	36.9	45.1	53.3	131
13 Constante de velocidad rpm / V	259	212	179	72.7
14 Relación velocidad / par rpm / mNm	4.26	5.44	5.85	3.82
15 Constante de tiempo mecánica ms	8.07	10.3	11.1	7.24
16 Inercia del rotor gcm²	181	181	181	181

Especificaciones

Datos térmicos

17	Resistencia térmica carcasa/ambiente	3.56 K/W
18	Resistencia térmica bobinado/carcasa	4.1 K/W
19	Constante de tiempo térmica del bobinado	29.6 s
20	Constante de tiempo térmica del motor	178 s
21	Temperatura ambiente	-40 ... +100°C
22	Máx. temperatura del bobinado	+125°C

Datos mecánicos (rodamiento a bolas pretensado)

23	Máx. velocidad permitida		10000 rpm
24	Juego axial con	< 4.0 N	0 mm
	carga axial	> 4.0 N	0.14 mm
25	Juego radial		pretensado
26	Carga axial máx. (dinámica)		3.8 N
27	Máx. fuerza de empuje a presión (estática)		50 N
	(idem, con eje sostenido)		1000 N
28	Carga radial máx. a 8 mm de la brida		21 N

Otras especificaciones

29	Número de pares de polos	8
30	Número de fases	3
31	Peso del motor	141 g

Los datos de la tabla son valores nominales.

Conexiones

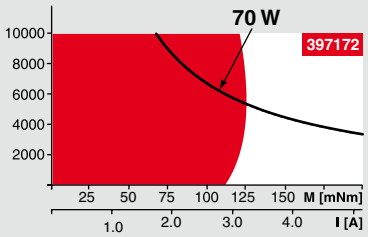
Pin 1	Sensor Hall 1*
Pin 2	Sensor Hall 2*
Pin 3	V_{Hall} 4.5 ... 18 VDC
Pin 4	Bobinado 3 motor
Pin 5	Sensor Hall 3*
Pin 6	GND
Pin 7	Bobinado 1 motor
Pin 8	Bobinado 2 motor

*interna pull-up (7 … 13 kΩ) su pin 3
Esquema de conexionado para los sensores Hall v. p. 35

Cables

Cable de la conexión Universal, L = 500 mm	**339380**
Cable de la conexión EPOS, L = 500 mm	**354045**

Rango de funcionamiento



Leyenda

Funcionamiento continuo
Teniendo en cuenta los valores de resistencia térmica antes mencionados (líneas 17 y 18). El rotor alcanzará la máxima temperatura durante funcionamiento continuo a 25°C de temperatura ambiente = límite térmico.

Funcionamiento intermitente
El motor puede ser sobrecargado durante cortos períodos (cíclicamente).

Potencia nominal asignada

Sistema Modular maxon

Esquema general en página 20–25

Reductor planetario
∅42 mm
3 - 15 Nm
Página 285

Reductor engranaje recto
∅45 mm
0.5 - 2.0 Nm
Página 286

Encoder MILE
256 - 2048 ppv,
2 canales
Página 308

Electrónicas Recomendadas:

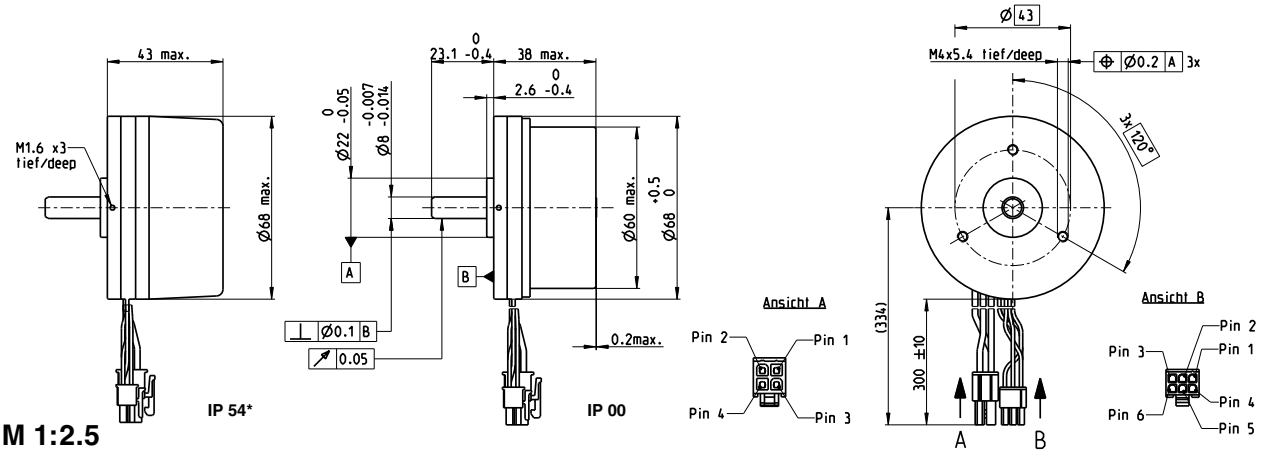
ESCON 36/3 EC	Página 342
ESCON Mod. 50/4 EC-S	343
ESCON Module 50/5	343
ESCON 50/5	344
DEC Module 50/5	346
EPOS2 Module 36/2	350
EPOS2 24/5, 50/5	351
EPOS2 P 24/5	354
EPOS3 70/10 EtherCAT	357
MAXPOS 50/5	360
Notas	**24**

Opción
Con Cables y Conector
(Temperatura ambiente -20 ... +100°C)

Edición julio 2014 / Sujeto a modificaciones

maxon flat motor

EC 60 flat ∅68 mm, Conmutación electrónica (Brushless), 100 W



M 1:2.5

Programa Stock
Programa Estándar
Programa Especial (previo encargo)

	Referencia					
IP 54* (con tapa)	412819		**408057**		412821	
IP 00 (sin tapa)		412823		**411678**		412825
Datos del motor						
Valores a tensión nominal						
1 Tensión nominal V	12	12	24	24	48	48
2 Velocidad en vacío rpm	3710	3710	4250	4250	3970	3970
3 Corriente en vacío mA	671	671	419	419	187	187
4 Velocidad nominal rpm	3260	3170	3840	3740	3580	3490
5 Par nominal (máx. par en continuo) mNm	231	279	227	289	257	319
6 Corriente nominal (máx. corriente en continuo) A	7.81	9.25	4.43	5.47	2.3	2.78
7 Par de arranque mNm	2850	2850	4180	4180	5010	5010
8 Corriente de arranque A	93.5	93.5	78.2	78.2	43.8	43.8
9 Máx. rendimiento %	84	84	86	86	88	88
Características						
10 Resistencia en bornes fase-fase Ω	0.128	0.128	0.307	0.307	1.1	1.1
11 Inductancia en bornes fase-fase mH	0.0615	0.0615	0.188	0.188	0.864	0.864
12 Constante de par mNm/A	30.5	30.5	53.4	53.4	114	114
13 Constante de velocidad rpm/V	313	313	179	179	83.4	83.4
14 Relación velocidad/par rpm/mNm	1.32	1.32	1.03	1.03	0.798	0.798
15 Constante de tiempo mecánica ms	16.7	16.7	13	13	10.1	10.1
16 Inercia del rotor gcm^2	1210	1210	1210	1210	1210	1210

Especificaciones

Datos térmicos

17	Resistencia térmica carcasa/ambiente		4.34 (2.5) K/W
18	Resistencia térmica bobinado/carcasa		3.5 K/W
19	Constante de tiempo térmica del bobinado		40 s
20	Constante de tiempo térmica del motor		155 (86.9) s
21	Temperatura ambiente		-40…+100°C
22	Máx. temperatura del bobinado		+125°C

Datos mecánicos (rodamiento a bolas pretensado)

23	Máx. velocidad permitida		6000 rpm
24	Juego axial con	< 12.0 N	0 mm
	carga axial	> 12.0 N	0.14 mm
25	Juego radial		pretensado
26	Carga axial máx. (dinámica)		12 N
27	Máx. fuerza de empuje a presión (estática)		170 N
	(idem, con eje sostenido)		8000 N
28	Carga radial máx. a 7.5 mm de la brida		100 N

Otras especificaciones

29	Número de pares de polos	
30	Número de fases	7
31	Peso del motor	3
	Los datos de la tabla son valores nominales.	470 g

Conexiones motor (cables AWG 18)

rojo	Bobinado 1 motor	Pin 1
negro	Bobinado 2 motor	Pin 2
blanco	Bobinado 3 motor	Pin 3
	N.C.	Pin 4

Conector	**N° de artículo**
Molex	39-01-2040

Conexiones sensores (cables AWG 28)

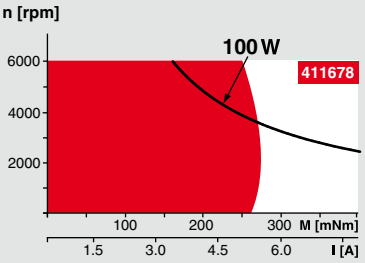
gris	Sensor Hall 1	Pin 1
gris	Sensor Hall 2	Pin 2
gris	Sensor Hall 3	Pin 3
gris	GND	Pin 4
azul	V_{Hall} 4.5…18 VDC	Pin 5
	N.C.	Pin 6

Conector	**N° de artículo**
Molex	430-25-0600

Esquema de conexionado para los sensores Hall ver página 35

* Grado de protección, solamente si está sellado por el lado de la brida.

Rango de funcionamiento



Leyenda

Funcionamiento continuo
Teniendo en cuenta los valores de resistencia térmica antes mencionados (líneas 17 y 18). El rotor alcanzará la máxima temperatura durante funcionamiento continuo a 25°C de temperatura ambiente = límite térmico.

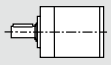
Funcionamiento intermitente
El motor puede ser sobrecargado durante cortos períodos (cíclicamente).

Potencia nominal asignada

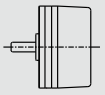
Sistema Modular maxon

Esquema general en página 20–25

Reductor planetario
∅52 mm
4 - 30 Nm
Página 288



Encoder MILE
512 - 4096 ppv,
2 canales
Página 309



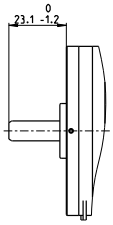
Electrónicas Recomendadas:

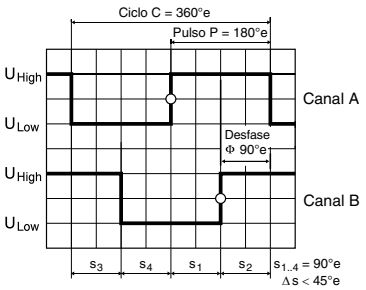
ESCON Mod. 50/5	Página 343
ESCON 50/5	344
ESCON 70/10	344
DEC Module 50/5	346
EPOS2 24/5	351
EPOS2 50/5	351
EPOS2 70/10	351
EPOS2 P 24/5	354
EPOS3 70/10 EtherCAT	357
MAXPOS 50/5	360
Notas	**24**

Edición julio 2014 / Sujeto a modificaciones

Encoder MILE 512–4096 ppv, 2 canales, con line driver

Integrado en el motor





Sentido de rotación cw (Definición cw P. 78)

maxon sensor

Programa Stock
Programa Estándar
Programa Especial (previo encargo)

Tipo	Referencia			
	421985	421986	421987	421988
Número de pulsos por vuelta	512	1024	2048	4096
Número de canales	2	2	2	2
Máx. frecuencia de funcionamiento (kHz)	500	500	500	500
Máx. velocidad (rpm)	6000	6000	6000	6000

Sistema Modular maxon

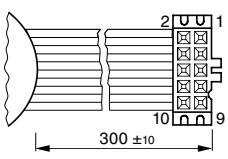
+ Motor	Página	+ Reductor	Página	+ Freno	Página	Longitud total [mm] / ● ver reductor			
EC 60 flat, IP00	236					39.0	39.0	39.0	39.0
EC 60 flat, IP00	236	GP 52, 4 - 30 Nm	288			●	●	●	●
EC 60 flat, IP54	236					43.0	43.0	43.0	43.0
EC 60 flat, IP54	236	GP 52, 4 - 30 Nm	288			●	●	●	●

Datos técnicos

Tensión de alimentación V_{CC}	5 V ± 10%
Señal de salida	CMOS y TTL compatible
Longitud de estado s_n (1000 rpm)	90°e ± <45°e
Tiempo del frente de subida (típico con C_L = 25 pF, R_L = 1 kΩ, 25 °C)	100 ns
Tiempo del frente de bajada (típico con C_L = 25 pF, R_L = 1 kΩ, 25 °C)	100 ns
Rango de temperaturas	-40…+100 °C
Momento de la inercia de la rueda de código	≤ 13 gcm²
Corriente de salida por canal	máx. 4 mA
Salida «open collector» de los sensores Hall con resistencia «pull-up»	10 kΩ ± 20% integrada
Esquema de conexionado para los sensores Hall	ver página 35

Más información en la sección «Descargas» de la tienda online.

Conexión

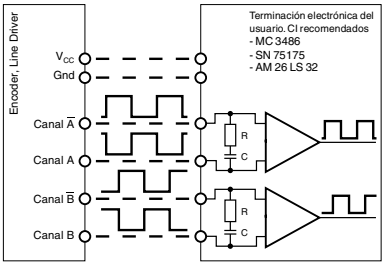


1	N.C.
2	V_{CC}
3	GND
4	N.C.
5	Canal $\overline{A}$
6	Canal A
7	Canal $\overline{B}$
8	Canal B
9	No conectar
10	No conectar

Conector DIN 41651/EN 60603-13
Cable plano AWG 28

Nota: No se permiten resistencias pull-down (< 100 kΩ) en las salidas del encoder. Se permiten resistencias pull-up, aunque no son necesarias.

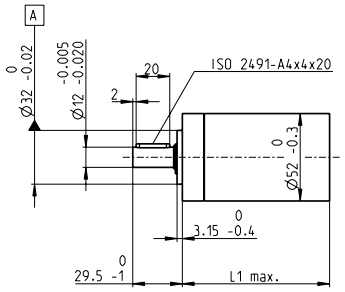
Ejemplo de conexión

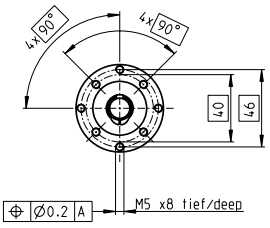


Resistencia de conexión R op. = típica 120 Ω
Condensador C ≥ 0.1 nF por cada metro lineal de línea

Reductor planetario GP 52 C ∅52 mm, 4–30 Nm

Versión cerámica





M 1:4

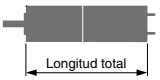
Datos técnicos

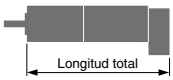
Reductor planetario	diente recto
Eje de salida	acero inoxidable
Rodamiento de salida	rodamiento a bolas pretensado
Juego radial a 12 mm de la brida	máx. 0.06 mm
Juego axial con carga axial < 5 N	0 mm
> 5 N	máx. 0.3 mm
Máx. carga axial admisible	200 N
Máx. fuerza adm. en acoplamientos a presión	500 N
Sentido de giro, entrada/salida	=
Máx. velocidad de entrada en continuo	6000 rpm
Rango de temperatura aconsejado	-15…+80°C
Rango de temp. extendido opcional	-40…+100°C
Número de etapas	1 2 3 4
Máx. carga radial adm. a 12 mm de la brida	420 N 630 N 900 N 900 N

maxon gear

Programa Stock
Programa Estándar
Programa Especial (previo encargo)

			Referencia						
			223080	223083	223089	223094	223097	223104	223109
Datos del Reductor									
1	Reducción		3.5:1	12:1	43:1	91:1	150:1	319:1	546:1
2	Reducción absoluta		7/2	49/4	343/8	91	2401/16	637/2	546
10	Momento de inercia	gcm²	20.7	17.6	17.3	16.7	17.3	16.8	16.4
3	Diámetro máx. del eje del motor	mm	10	10	10	10	10	10	10
	Referencia		**223081**	223084	223090	223095	223099	**223105**	223110
1	Reducción		4.3:1	15:1	53:1	113:1	186:1	353:1	676:1
2	Reducción absoluta		13/3	91/6	637/12	338/3	4459/24	28561/81	676
10	Momento de inercia	gcm²	12	16.8	17.2	9.3	17.3	9.4	9.1
3	Diámetro máx. del eje del motor	mm	8	10	10	8	10	8	8
	Referencia			**223085**	223091	223096	223101	223106	223111
1	Reducción			19:1	66:1	126:1	230:1	394:1	756:1
2	Reducción absoluta			169/9	1183/18	126	8281/36	1183/3	756
10	Momento de inercia	gcm²		9.5	16.7	16.4	16.8	16.7	16.4
3	Diámetro máx. del eje del motor	mm		8	10	10	10	10	10
	Referencia			223086	223092	223098	223102	223107	223112
1	Reducción			21:1	74:1	156:1	257:1	441:1	936:1
2	Reducción absoluta			21	147/2	156	1029/4	441	936
10	Momento de inercia	gcm²		16.5	17.2	9.1	17.3	16.5	9.1
3	Diámetro máx. del eje del motor	mm		10	10	8	10	10	8
	Referencia			223087	**223093**		223103	223108	
1	Reducción			26:1	81:1		285:1	488:1	
2	Reducción absoluta			26	2197/27		15379/54	4394/9	
10	Momento de inercia	gcm²		9.1	9.4		16.7	9.4	
3	Diámetro máx. del eje del motor	mm		8	8		10	8	
4	Número de etapas		1	2	3	3	4	4	4
5	Máx. par en continuo	Nm	4	15	30	30	30	30	30
6	Máx. par admisible de forma intermitente	Nm	6	22.5	45	45	45	45	45
7	Máx. rendimiento	%	91	83	75	75	68	68	68
8	Peso	g	460	620	770	770	920	920	920
9	Holgura media en vacío	°	0.6	0.8	1.0	1.0	1.0	1.0	1.0
11	Longitud reductor L1	mm	49.0	65.0	78.5	78.5	92.0	92.0	92.0



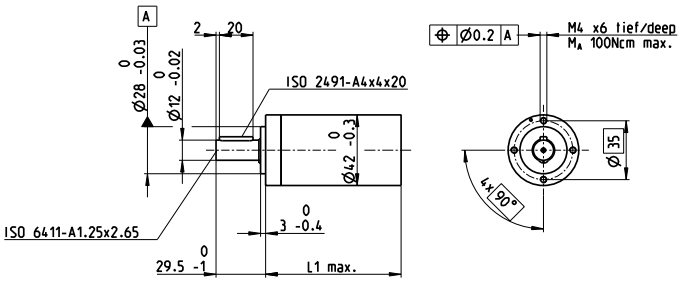


Sistema Modular maxon

+ Motor	Página	+ Sensor	Página	Freno	Pág.	Longitud total [mm] = Longitud motor + longitud reductor + (sensor/freno) + piezas de montaje						
RE 40, 150 W	114					120.1	136.1	149.6	149.6	163.1	163.1	163.1
RE 40, 150 W	114	MR	320			131.5	147.5	161.0	161.0	174.5	174.5	174.5
RE 40, 150 W	114	HED_ 5540	325/327			140.8	156.8	170.3	170.3	183.8	183.8	183.8
RE 40, 150 W	114	HEDL 9140	331			174.1	190.1	203.6	203.6	217.1	217.1	217.1
RE 40, 150 W	114			AB 28	372	156.2	172.2	185.7	185.7	199.2	199.2	199.2
RE 40, 150 W	114			AB 28	373	164.2	180.2	193.7	193.7	207.2	207.2	207.2
RE 40, 150 W	114	HED_ 5540	325/327	AB 28	372	173.4	189.4	202.9	202.9	216.4	216.4	216.4
RE 40, 150 W	114	HEDL 9140	331	AB 28	373	184.6	200.6	214.1	214.1	227.6	227.6	227.6
RE 50, 200 W	115					157.1	173.1	186.6	186.6	200.1	200.1	200.1
RE 50, 200 W	115	HED_5540	326/328			177.8	193.8	207.3	207.3	220.8	220.8	220.8
RE 50, 200 W	115	HEDL 9140	332			219.5	235.5	249.0	249.0	262.5	262.5	262.5
RE 50, 200 W	115			AB 44	376	219.5	235.5	249.0	249.0	262.5	262.5	262.5
RE 50, 200 W	115	HEDL 9140	332	AB 44	376	232.5	248.5	262.0	262.0	275.5	275.5	275.5
EC 40, 170 W	193					129.1	145.1	158.6	158.6	172.1	172.1	172.1
EC 40, 170 W	193	HED_5540	326/328			152.5	168.5	182.0	182.0	195.5	195.5	195.5
EC 40, 170 W	193	Res 26	337			156.3	172.3	185.8	185.8	199.3	199.3	199.3
EC 40, 170 W	193			AB 32	374	171.8	187.8	201.3	201.3	214.8	214.8	214.8
EC 40, 170 W	193	HED_5540	326/328	AB 32	374	190.2	206.2	219.7	219.7	233.2	233.2	233.2

Reductor planetario GP 42 C ∅42 mm, 3–15 Nm

Versión cerámica



M 1:4

Datos técnicos

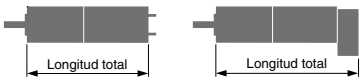
Reductor planetario	diente recto
Eje de salida	acero inoxidable
Rodamiento de salida	rodamiento a bolas pretensado
Juego radial a 12 mm de la brida	máx. 0.06 mm
Juego axial con carga axial < 5 N	0 mm
> 5 N	máx. 0.3 mm
Máx. carga axial admisible	150 N
Máx. fuerza adm. en acoplamientos a presión	300 N
Sentido de giro, entrada/salida	=
Máx. velocidad de entrada en continuo	8000 rpm
Rango de temperatura aconsejado	-40...+100°C
Número de etapas	1 2 3 4
Máx. carga radial adm. a 12 mm de la brida	120 N 240 N 360 N 360 N

maxon gear

Programa Stock
Programa Estándar
Programa Especial (previo encargo)

Referencia		203113	203115	203119	203120	203124	203129	203128	203133	203137	203141
Datos del Reductor											
1 Reducción		3.5:1	12:1	26:1	43:1	81:1	156:1	150:1	285:1	441:1	756:1
2 Reducción absoluta		7/2	49/4	26	343/8	2197/27	156	2401/16	15379/54	441	756
10 Momento de inercia	gcm²	14	15	9.1	15	9.4	9.1	15	15	14	14
3 Diámetro máx. del eje del motor	mm	10	10	8	10	8	8	10	10	10	10
Referencia		203114	203116	260552*	203121	203125	260553*	203130	203134	203138	203142
1 Reducción		4.3:1	15:1	36:1	53:1	91:1	216:1	186:1	319:1	488:1	936:1
2 Reducción absoluta		13/3	91/6	36/1	637/12	91	216/1	4459/24	637/2	4394/9	936
10 Momento de inercia	gcm²	9.1	15	5.0	15	15	5.0	15	15	9.4	9.1
3 Diámetro máx. del eje del motor	mm	8	10	4	10	10	4	10	10	8	8
Referencia		260551*	203117		203122	203126		203131	203135	203139	260554*
1 Reducción		6:1	19:1		66:1	113:1		230:1	353 :1	546:1	1296:1
2 Reducción absoluta		6/1	169/9		1183/18	338/3		8281/36	28561/81	546	1296/1
10 Momento de inercia	gcm²	4.9	9.4		15	9.4		15	9.4	14	5.0
3 Diámetro máx. del eje del motor	mm	4	8		10	8		10	8	10	4
Referencia			203118		203123	203127		203132	203136	203140	
1 Reducción			21:1		74:1	126:1		257:1	394:1	676:1	
2 Reducción absoluta			21		147/2	126		1029/4	1183/3	676	
10 Momento de inercia	gcm²		14		15	14		15	15	9.1	
3 Diámetro máx. del eje del motor	mm		10		10	10		10	10	8	
4 Número de etapas		1	2	2	3	3	3	4	4	4	4
5 Máx. par en continuo	Nm	3.0	7.5	7.5	15.0	15.0	15.0	15.0	15.0	15.0	15.0
6 Máx. par admisible de forma intermitente	Nm	4.5	11.3	11.3	22.5	22.5	22.5	22.5	22.5	22.5	22.5
7 Máx. rendimiento	%	90	81	81	72	72	72	64	64	64	64
8 Peso	g	260	360	360	460	460	460	560	560	560	560
9 Holgura media en vacío	°	0.6	0.8	0.8	1.0	1.0	1.0	1.0	1.0	1.0	1.0
11 Longitud reductor L1	mm	41.0	55.5	55.5	70.0	70.0	70.0	84.5	84.5	84.5	84.5

*no combinación con motor EC 45 (150 W y 250 W)



Sistema Modular maxon

+ Motor	Página	+ Sensor	Pág.	Freno	Pág.	Longitud total [mm] = Longitud motor + longitud reductor + (sensor/freno) + piezas de montaje									
RE 35, 90 W	112					112.1	126.6	126.6	141.1	141.1	141.1	155.6	155.6	155.6	155.6
RE 35, 90 W	112	MR	320			123.5	138.0	138.0	152.5	152.5	152.5	167.0	167.0	167.0	167.0
RE 35, 90 W	112	HED_ 5540	325/327			132.8	147.3	147.3	161.8	161.8	161.8	176.3	176.3	176.3	176.3
RE 35, 90 W	112	DCT 22	336			130.2	144.7	144.7	159.2	159.2	159.2	173.7	173.7	173.7	173.7
RE 35, 90 W	112			AB 28	372	148.2	162.7	162.7	177.2	177.2	177.2	191.7	191.7	191.7	191.7
RE 35, 90 W	112	HED_ 5540	325/327	AB 28	372	165.4	179.9	179.9	194.4	194.4	194.4	208.9	208.9	208.9	208.9
RE 40, 150 W	114					112.1	126.6	126.6	141.1	141.1	141.1	155.6	155.6	155.6	155.6
RE 40, 150 W	114	MR	320			123.5	138.0	138.0	152.5	152.5	152.5	167.0	167.0	167.0	167.0
RE 40, 150 W	114	HED_ 5540	325/327			132.8	147.3	147.3	161.8	161.8	161.8	176.3	176.3	176.3	176.3
RE 40, 150 W	114	HEDL 9140	331			166.2	180.7	180.7	195.2	195.2	195.2	209.7	209.7	209.7	209.7
RE 40, 150 W	114			AB 28	372	148.2	162.7	162.7	177.2	177.2	177.2	191.7	191.7	191.7	191.7
RE 40, 150 W	114			AB 28	373	156.2	170.7	170.7	185.2	185.2	185.2	199.7	199.7	199.7	199.7
RE 40, 150 W	114	HED_ 5540	325/327	AB 28	372	165.4	179.9	179.9	194.4	194.4	194.4	208.9	208.9	208.9	208.9
RE 40, 150 W	114	HEDL 9140	331	AB 28	373	176.7	191.2	191.2	205.7	205.7	205.7	220.2	220.2	220.2	220.2
EC 40, 170 W	193					121.1	135.6	135.6	150.1	150.1	150.1	164.6	164.6	164.6	164.6
EC 40, 170 W	193	HED_ 5540	326/328			144.5	159.0	159.0	175.5	175.5	175.5	188.0	188.0	188.0	188.0
EC 40, 170 W	193	Res 26	337			148.3	162.8	162.8	177.3	177.3	177.3	191.8	191.8	191.8	191.8
EC 40, 170 W	193			AB 32	374	163.8	178.3	178.3	192.8	192.8	192.8	207.3	207.3	207.3	207.3
EC 40, 170 W	193	HED_ 5540	326/328	AB 32	374	182.2	196.7	196.7	211.2	211.2	211.2	225.7	225.7	225.7	225.7
EC 45, 150 W	194					152.3	166.8	166.8	181.3	181.3	181.3	195.8	195.8	195.8	195.8
EC 45, 150 W	194	HEDL 9140	331			167.9	182.4	182.4	196.9	196.9	196.9	211.4	211.4	211.4	211.4
EC 45, 150 W	194	Res 26	337			152.3	166.8	166.8	181.3	181.3	181.3	195.8	195.8	195.8	195.8
EC 45, 150 W	194			AB 28	373	159.7	174.2	174.2	188.7	188.7	188.7	203.2	203.2	203.2	203.2
EC 45, 150 W	194	HEDL 9140	331	AB 28	373	176.7	191.2	191.2	205.7	205.7	205.7	220.2	220.2	220.2	220.2
EC 45, 250 W	195					185.1	199.6	199.6	214.1	214.1	214.1	228.6	228.6	228.6	228.6
EC 45, 250 W	195	HEDL 9140	331			200.7	215.2	215.2	229.7	229.7	229.7	244.2	244.2	244.2	244.2
EC 45, 250 W	195	Res 26	337			185.1	199.6	199.6	214.1	214.1	214.1	228.6	228.6	228.6	228.6
EC 45, 250 W	195			AB 28	373	192.5	207.0	207.0	221.5	221.5	221.5	236.0	236.0	236.0	236.0
EC 45, 250 W	195	HEDL 9140	331	AB 28	373	209.5	224.0	224.0	238.5	238.5	238.5	253.0	253.0	253.0	253.0

Referencias

[1] Fundamentos de Robótica. Antonio Barrientos. MCGraw-Hill

[2] Fundamentals of Robotics. Analisys and Control. R.J. Schilling. Prentice-Hall

[3] Karlj, A.R. and Bajd, T. (1989). Functional electrical stimulation: standing and walking after spinal cord injury. Boca Raton, Fla : CRC Press, c1989

[4] Robótica: control, detección, visión e inteligencia. K.S. Fu, R.C. González y C.S.G. Lee. McGraw-Hill

[5] Página Oficial de EXOLEGS, `http://www.exo-legs.org`

[6] Página Oficial de Maxon, `http://www.maxonmotor.es`

[7] Página Oficial de Matlab, `http://es.mathworks.com`

[8] Página Oficial de Cyberdine, `http://www.cyberdyne.jp`

[9] Página Oficial de Rex Bionics, `http://www.rexbionics.com`

[10] Página Oficial de Ekso Bionics, `http://intl.eksobionics.com`